ISBN : 979-3757-23-9

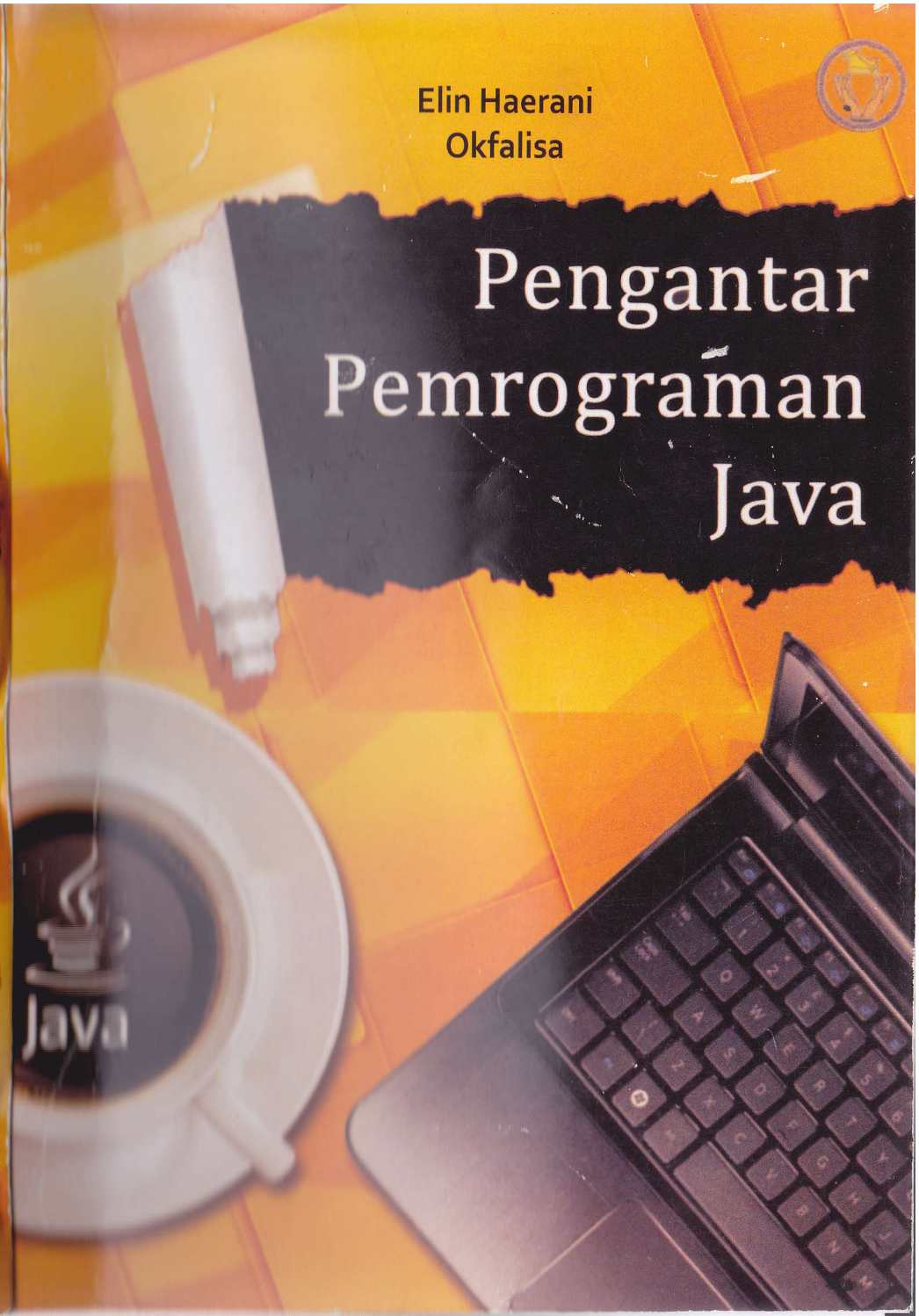
Pengantar Pemrograman Java

Elin Haerani

Pengantar Pemrograman Java

Elin Haerani

Okfalisa



# Pengantar Pemrograman Java

Elin Haerani, ST, M.Kom
Dr. Okfalisa, St, M.Sc

# Pengantar Pemrograman Java

Daulat Riau
2013

## KATA PENGANTAR

Terima Kasih kepada Allah SWT, buku daras Pengantar Pemrograman Java ini dapat terselesaikan seperti yang diharapkan.

Buku ini terdiri dari 16 Bab, yang tiap bab saling berkaitan satu dengan yang lainnya. Buku ini dibuat untuk pengguna awam yang akan mempelajari bahasa Java, pada buku ini akan dikenalkan dengan bahasa java dengan editor **NetBeans** dan TextPad yang akan membantu didalam pembuatan program.

Penulis yakin masih banyak kekurangan dari buku ini, saran dan kritik sangat penulis harapkan.

Pekanbaru, Juli 2013

Penulis

## DAFTAR ISI

# BAB 1

# ALGORITMA DAN PEMROGRAMAN DENGAN NETBEANS

## 1. TEORI SINGKAT

### 1.1. Pengenalan Algoritma

Algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis. Kata logis disini berarti benar sesuai dengan logika manusia. Untuk menjadi sebuah algoritma, urutan langkah yang ditempuh untuk menyelesaikan masalah harus memberikan hasil yang benar.

Misalkan saja "algoritma aktifitas pagi hari sebelum berangkat kerja" yang dikerjakan oleh seorang eksekutif junior untuk turun dari tempat tidur dan bekerja: (1) Turun dari tempat tidur; (2) melepas piyama; (3) mandi; (4) berpakaian; (5) makan pagi; (6) baca koran; (7) pergi kerja. Rutin ini membuat eksekutif junior bekerja dengan persiapan yang baik untuk membuat

keputusan-keputusan kritis. Andaikan langkah yang sama dilakukan dengan urutan yang sedikit berbeda, misalnya: (1) Turun dari tempat tidur; (2) melepas piyama; (3) berpakaian; (4) mandi; (5) baca koran; (6) makan pagi; (7) berangkat kerja. Di dalam kasus ini eksekutif junior tersebut berangkat ke tempat kerja dalam keadaan basah kuyub.

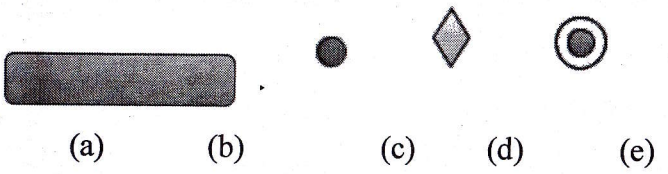
## 1.2. pengenalan UML

*Unified Modeling Language* (UML) belakangan ini merupakan skema representasi grafis yang banyak digunakan secara luas untuk pemodelan sistem berorientasi object. UML ini telah menyatukan berbagai skema notasi populer bersama-sama. Banyak yang merancang sistem mengunakan bahasa ini (dalam bentuk diagram) untuk memodelkan sistem mereka.

UML adalah bahasa grafis yang kompleks dan kaya dengan fitur. Salah satu model diagramnya adalah *Activity Diagram*. Dalam pembahasan algoritma ini, digram UML yang akan digunakan adalah *activity diagram*.



Sebuah *activity diagram* memodelkan aspek dari tingkah laku sistem. *Activity diagram* memodel aliran kerja obyek (urutan aktivitas) selama eksekusi program. *Activity diagram* adalah flowchart yang memodel aksi yang akan dikerjakan oleh obyek beserta dengan ordenya.



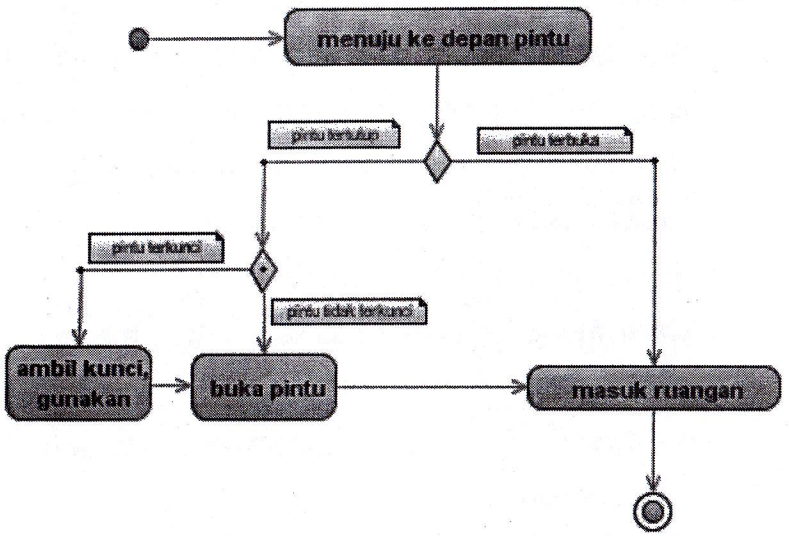

Gambar 1.1. Simbol *Activity Diagram*

UML merepresentasikan aktivitas dengan simbol sebuah oval (Gambar 1.a.) di dalam *activity diagram*. Nama aktivitas diletakkan di dalam oval. Sebuah anak panah (Gambar 1.b.) menghubungkan dua aktivitas yang menunjukkan orde dimana aktivitas dikerjakan. Lingkaran padat (Gambar 1.c.) menunjukkan dimulainya aktivitas. Percabangan ditunjukkan dengan sebuah jajaran

genjang (Gambar 1.d.) dan lingkaran padat dan ditambah dengan lingkaran di luarnya menandakan akhir dari aktifitas (Gambar 1.e.).

Sekarang, mari kita lihat contoh *activity diagram*. Misalkan kita akan membuat *activity diagram* seseorang yang akan masuk ke dalam sebuah ruangan yang berpintu. Pertama kali yang dikerjakan adalah menuju ke pintu. Kemudian melihat apakah pintu dalam keadaan terbuka atau tertutup. Jika dalam keadaan terbuka, dia langsung masuk. Jika pintu dalam keadaan tertutup, maka orang tersebut akan mengecek apakah pintu dalam keadaan terkunci atau tidak. Kalau terkunci, maka orang tersebut akan mengambil kunci dan membuka pakai kunci, jika tidak terkunci, dia akan langsung membuka pintu. Dengan orang tersebut sudah bisa masuk ke ruangan, maka aktifitas masuk ruangan selesai. *Activity diagram* untuk proses tersebut diperlihatkan pada Gambar 1.2.






Gambar 1.2. *Activity diagram* untuk masuk ruangan

## 1.3. Pengenalan NetBeans

NetBeands adalah merupakan IDE yang ditujukan untuk memudahkan pemrograman java. Dalam NetBeans , pemrograman dilakukan berbasis visual dan event driven. Persis seperti IDE lain, misalnya Borland Delphi dan Micrsoft Visual Studio.

Untuk membuat dialog atau user-interface, kita tidak perlu membuat teks program secara manual baris per baris,tetapi cukup klik pada component-

pallete. Teks program akan dihasilkan secara otomatis. NetBeans mencakup compiler atau builder , dan debugger internal. Hal ini sangat memudahkan proses paska perancangan program. Proses deployment dan atau tanpa tes dapat juga dilakukan dari dalam NetBeans.

Cara menggunakan NetBeans untuk menjalankan java.

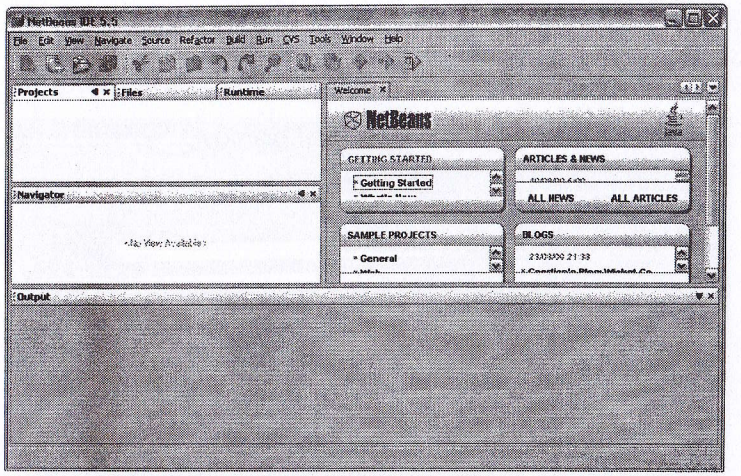
1. Pastikan dulu anda sudah menginstal java di computer Anda
2. Kemudian download juga NetBeans
3. Instal NetBeans di komputerta Anda
4. Setelah kedua software siap, jalankan NetBeans

## 2. LATIHAN INSTALASI JAVA

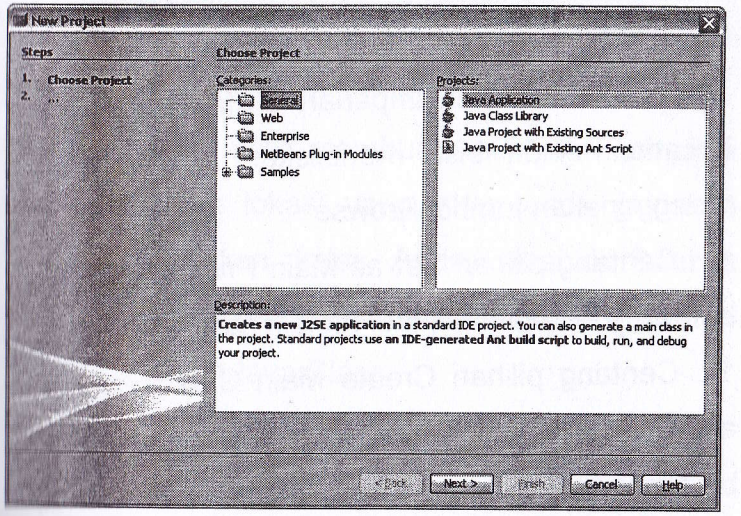
Langkah – lahkah Latihan

1. Pilih Start → Program →NetBeands 5.5
2. Anda akan peroleh tampilan awal sebagai berikut :



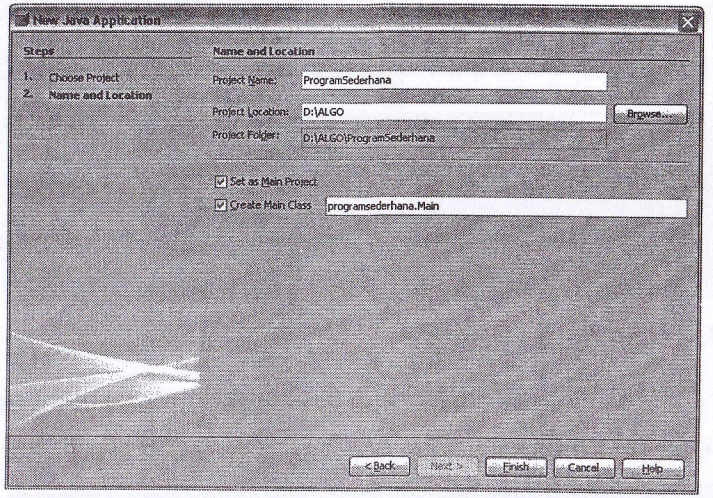



3. Pilih menu File
4. Pilih sub menu New Project. Akan muncul layar sebagai berikut




5. Pilih General, pada pilihan Categories

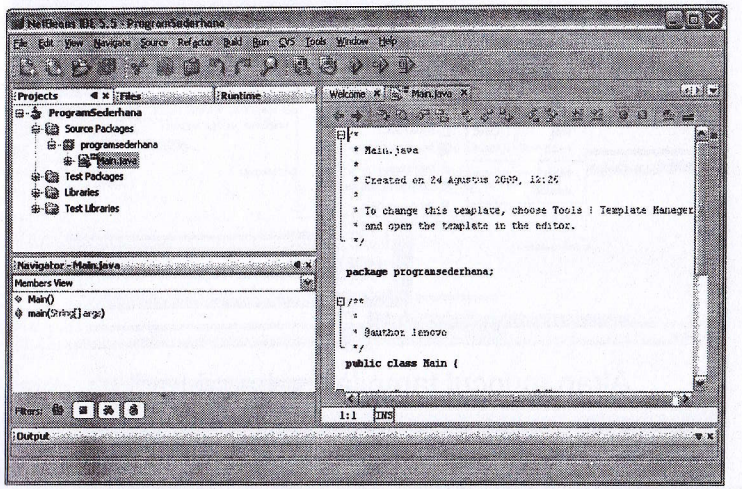
6. Pilih Java Aplication pada pilihan Project
7. Klik Next >
8. Akan muncul layar sebagai berikut



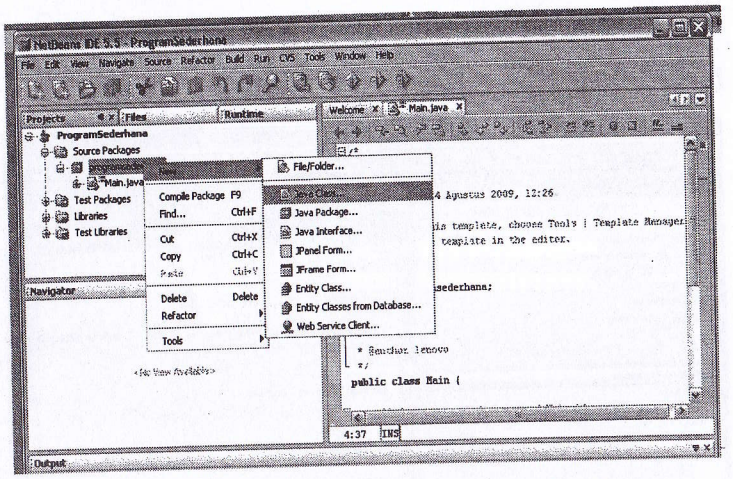

9. Tulis nama project pada isian Project Name
10. Pilih lokasi penyimpanan pada isian Project Location. Anda bisa menuliskan lokasinya atau menggunakan tombol Browse
11. Centang pilihan Set as Main Project, jika Anda berharap ini akan menjadi project utama Anda.
12. Centang pilihan Create Main Class jika Anda akan langsung membuat kelas main. Atau Anda bisa mengganti nama kelasnya(bukan kelas main)



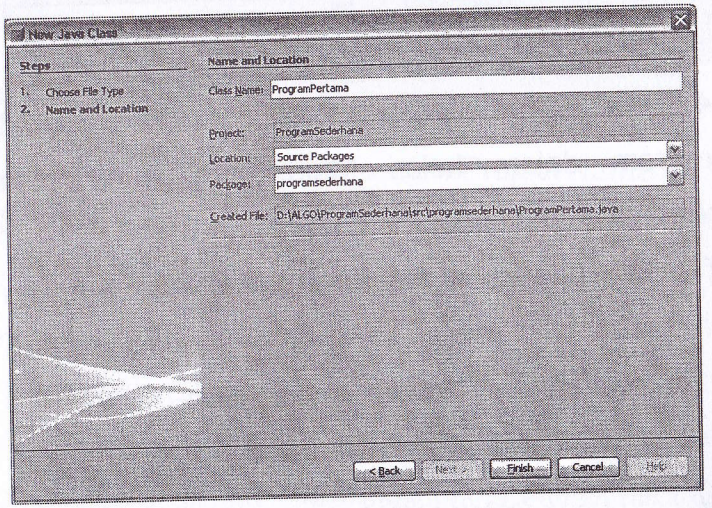
13. Kemudian klik finish. Anda akan menjumpai tampilan sebagi berikut:




14. Setelah tampilan ini, Anda bisa membuat program dalam kelas Main atau Anda bisa membuat kelas sendiri.
15. Jika anda akan membuat kelas dari tampilan ini, maka pilih lokasi yang akan anda gunakan untuk meletakkan kelas Anda, bisa di **Source Package** atau programsederhana (dalam contoh ini). Pilih New→Java Class



16. Akan muncul tampilan sebagai berikut:

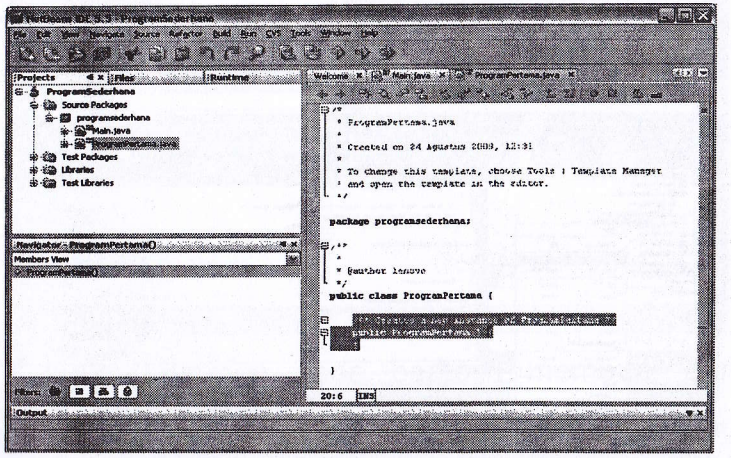


17. Kemudian tuliskan nama kelasnya. Dalam contoh ini diberi nama ProgramPertama

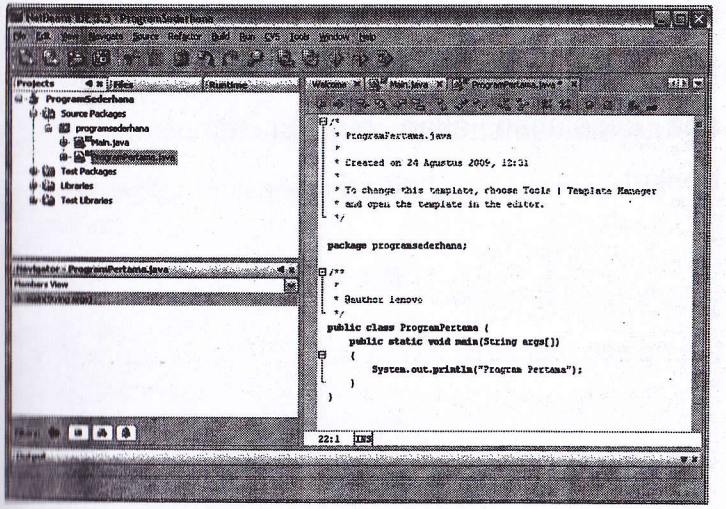
18. Klik Finish



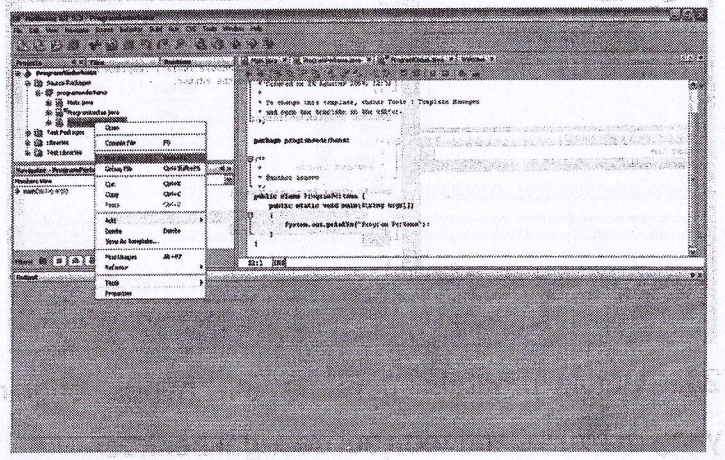
19. Anda akan menjumpai tampilan seperti berikut:



20. Untuk program pertama ini, hapus bagian yang di blok. Ganti menjadi sebagai berikut:

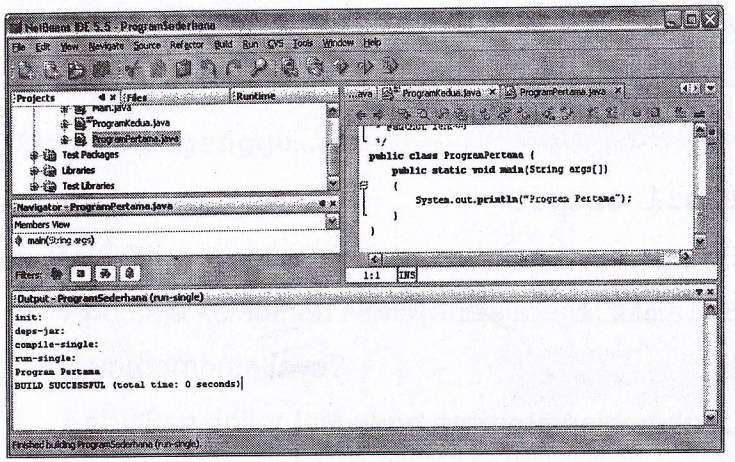


21. Setelah itu jalankan dengan cara klik kanan pada tab **Projects** pada nama kelasnya, dalam hal ini ProgramPertama. Tampilannya adalah sebagi berikut:



22. pilih run Akan terjadi proses kompilasi. Perhatikan bagian tab Output. Jika berhasil, tidak ada kesalahan akan muncul tampilan sebagai berikut :






Dengan demikian anda berhasil membuat dan menjalankan program java dengan menggunakan NetBeans.

## 3. SOAL LATIHAN

a) Buatlah project baru dengan nama Algoritma1

b) Buatlah kelas baru dengan nama Latihan1

c) Ketikkan program berikut :

```
public class Latihan1
{
public static void main(String args[])
{
// ini adalah komentar
     System.out.println("Halooo
semua...");
```

```
System.out.println("Selamat     belajar
java");
}
}
```

**Hasil Output**

```
Halooo semua...
Selamat belajar java
```

Ubah bagian program berikut

```
System.out.println("Halooo semua...");
System.out.println("Selamat belajar
java");
```

**Menjadi**

```
System.out.print("Halooo semua...");
System.out.print("Selamat belajar
java");
```

d) Dari hasil output apa yang dapat Anda simpulkan?

e) Buatlah kelas baru lagi dengan nama Latihan2

f) Buatlah program untuk menampilkan output seperti berikut :



**Output**

```
Kami sedang belajar java
Jangan diganggu...!
```

## 4. TUGAS

1. Apa saja kelebihan NetBeans sebagai IDE pemrograman java?
2. Sebutkan editor lain yang dapat digunakan untuk program java!

# BAB 2

# PEMROGRAMAN DENGAN JAVA

## 1. TEORI SINGKAT

Mengapa Java? Karena java adalah bahasa pemrograman multi platform. Java tidak menyediakan IDE khusus seperti halnya bahasa pemrograman yang lain. Pemrogram bisa menggunakan IDE yang *support* ke Java, misalnya Netbeans, Eclips, TexPad, dan lain-lain. Editor teks bisa digunakan semisal Notepad. Jika editor yang digunakan tidak support Java, kita tinggal menyimpannya dalam ekstensi .java kemudian kompilasi dan menjalankannya menggunakan command prompt.

Secara umum, elemen-elemen dasar pemrograman Java terdiri dari :

**1. Himpunan Karakter**



Himpunaan karakter terdiri dari huruf, digit maupun simbol-simbol lainya (termasuk spasi, karakter kontrol).

Contoh :

Huruf : A, a, B, b, C, c .....

Digit : 0, 1, 2, 3, 4, 5 .....

Simbol dan lainnya : _ - + * dan sebagainya

**2. Pengenal (identifier)**

Pengenal atau identifier adalah suatu nama yang bisa dipakai dalam pemrograman untuk menyatakan :

- variabel
- konstanta bernama
- tipe data
- fungsi
- label
- obyek

Contoh :

moMhs;

no_Mhs;

### 3. Kata Kunci

Pengenal sistem yang mempunyai makna khusus bagi kompiler. Kegunaan dari golongan ini tidak dapat diubah.

Contoh :

case, char, const, do, else, for, return, void, while, dan lain-lain

### 4. Tipe Data Primitif

Bahasa Pemrograman Java adalah bahasa pemrograman yang selalu menggunakan tipe data untuk setiap variabelnya. Itu berarti bahwa semua variabel harus dideklarasikan terlebih dahulu sebelum mereka digunakan.

Misalnya

```
int nilai = 1;
```

- **byte**: Tipe data byte adalah 8-bit integer bertanda *two's complement*. Tipe ini mempunyai nilai minimum -128 dan nilai maksimumnya adalah 127.



- **short**: Tipe data short merupakan integer 16 bit *two's complement* yang mempunyai nilai minimum -32,768 dan nilai maksimum 32,767.
- **int**: Tipe data int adalah integer 32 bit *two's complement*. Dia mempunyai nilai minimum -2,147,483,648 sedangkan nilai maksimumnya adalah 2,147,483,647 (inclusive).
- **long**: Tipe data long adalah integer 64 bit *two's complement*. Nilai minimumnya adalah -9,223,372,036,854,775,808 sedangkan nilai maksimumnya adalah 9,223,372,036,854,775,807 (inclusive). Gunakan tipe data ini pada saat anda memerlukan jangkau nilai yang lebih besar daripada yang bisa disajikan oleh int.

- **float**: Tipe data float merupakan *single-precision 32-bit IEEE 754 floating point.*
- **double**: Tipe data double adalah *double-precision 64-bit IEEE 754 floating point.*

Tabel 1.1. Batas nilai floating point

| Parameter | float | double |
|---|---|---|
| N | 24 | 53 |
| K | 8 | 11 |
| $E_{max}$ | +127 | +1023 |
| $E_{min}$ | -126 | -1022 |

- **boolean**: Tipe data boolean hanya mempunyai dua nilai yang mungkin, yaitu true dan false. Gunakan tipe data ini untuk flag-flag sederhana untuk menjejak/menelusuri kondisi true atau false. Tipe data ini merepresentasikan satu bit informasi, tetapi ukurannya tidak didefinisikan dengan tepat.
- **char**: Tipe data char adalah karakter Unicode 16 bit. Tipe data ini mempunyai nilai minimum '\u0000' (atau 0) dan nilai maksimum '\uffff' (atau 65,535).

## 5. Variabel dan Konstanta

Variabel digunakan dalam program untuk menyimpam suatu nilai, dan nilai yang ada padanya dapat diubah selama ekseskusi berlangsung.

Konstanta adalah nilai yang tetap.



Contoh :

```
float jumlah;
jumlah = 10;
```

## 6. Konstanta bernama

Hal ini dapat dilakukan dengan menggunakan kata kunci const.

Contoh :

```
const float PHI = 3.14;
```

## 2. NILAI DEFAULT

Tidak selalu perlu untuk menentukan suatu nilai ketika sebuah field dideklarasikan. Field yang dideklarasikan tetapi tidak diinisialisasi akan diset ke default yang ada oleh kompiler. Secara umum, default ini akan bernilai *null* atau *zero* tergantung pada tipe datanya. Tabel 1.2 merangkum nilai default untuk tipe-tipe data di atas.

Tabel 1.2. Daftar nilai default untuk tipe data tertentu

| Tipe data | Nilai Default (untuk field) |
|---|---|
| byte | 0 |
| short | 0 |
| int | 0 |
| long | 0L |
| float | 0.0f |
| double | 0.0d |
| char | '\u0000' |
| String (atau obyek) | Null |
| boolean | False |

## 3. PERNYATAAN MASUKAN DAN KELUARAN DALAM JAVA

Dalam bahasa pemrograman Java, kita akan menggunakan pernyataan masukan dan keluaran. Pernyataan masukan adalah pernyataan untuk mendapatkan masukan dari keyboard. Sedangkan pernyataan keluaran adalah pernyataan untuk menampilkan sesuatu nilai ke layar.

1. Komponen keluaran

Untuk menampilkan ke layar secara tekstual, java mempunyai fasilitas output, yaitu dengan perintah



System.out.print( *hal yang akan ditampilkan, bisa lebih dari satu*);

untuk menampilkan ke layar dan setelah selesai tidak berpindah baris. Akibatnya perintah keluaran berikutnya akan ditampilkan pada baris yang sama.

Dan perintah

System.out.println(*hal yang akan ditampilkan, bisa lebih dari satu*);

untuk menampilkan ke layar dan setelah selesai berpindah baris. Akibatnya perintah keluaran setelah itu akan ditampilkan pada baris berikutnya.

System.out.println("Hasil penjumlahan "+hasil);
//menampilkan isi variabel hasil

2. Komponen masukan

Untuk memasukkan sebuah nilai ke variabel yang sudah didefinisikan digunakan kelas Scanner (kelas ini disediakan mulai Java versi 1.5).

**a. Input data bertipe Integer**

Untuk menginputkan data dengan tipe integer digunakan method nextInt di dalam kelas Scanner.

**b. Input data bertipe String**

Untuk menginputkan data dengan tipe integer digunakan method nextInt di dalam kelas Scanner.

**c. Input tipe yang lain**

Untuk input data dari keyboard dengan tipe yang lain dan disediakan oleh kelas Scannr adalah sebagai berikut

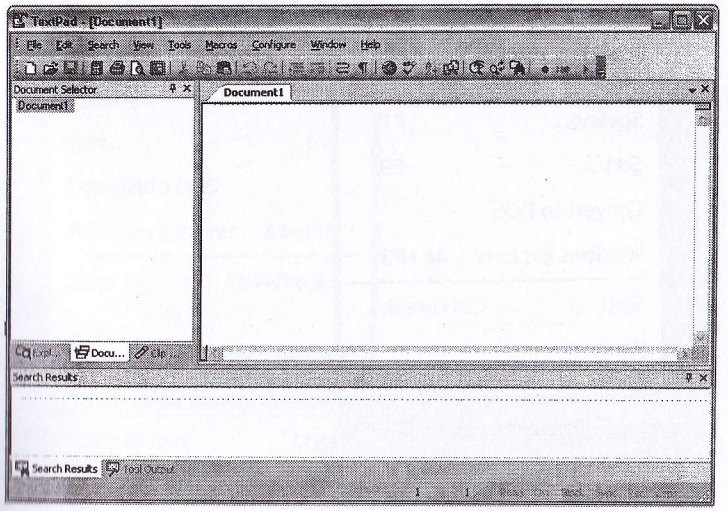
- nextBoolean : input tipe boolean (true atau false saja)
- nextShort : input tipe short integer
- nextLong : input tipe long integer
- nextFloat : input tipe float
- nextDouble : input tipe double

## 2. LATIHAN MEMBUAT PROGRAM SEDERHANA

Mari kta mulai mencoba mempraktekan program sedehana, dengan menggunakan TextPad. TextPad adalah salah satu editor yang dapat digunakan untuk pemrograman Java. Langkah-langkah menggunakan TextPad :



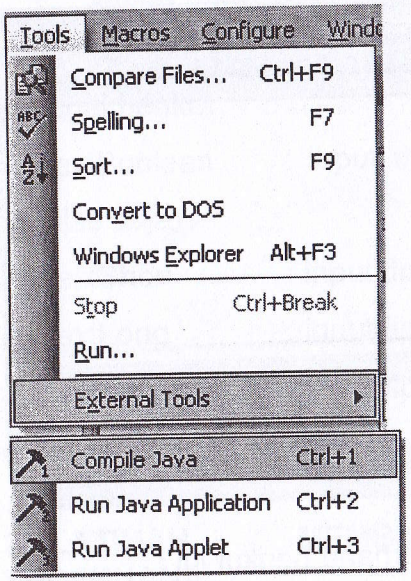
1. Pilih Menu Start
2. Pilih Programming
3. Pilih TextPad, maka akan muncul tampilan awal TextPad sebagai berikut :




4. Ketik program sederhana berikut ini :

```
public class Pertama
{
public static void main(String args[])
{
  System.out.println("Selamat Datang
di UIN SUSKA RIAU");
}
}
```

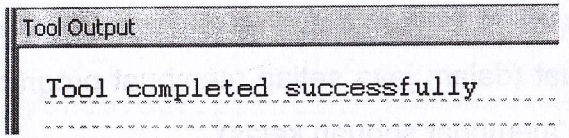
5. Simpan dengan nama **Pertama harus sama dengan nama class** yang ada pada program
6. Jalankan program tersebut dengan memilih menu sbb:



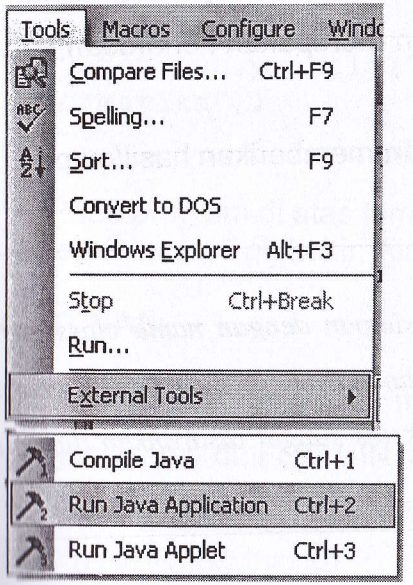

7. Setelah tidak ada kesalahan dan pada Tools Output muncul tulisan sebagai berikut :

Jika tidak ada pesan kesalahan, berarti kompilasi berhasil. Ini ditandai dengan adanya tambahan satu file yaitu **Pertama.class**.






8. Lakukan perintah berikut :




9. Maka akan muncul hasil output sebagai berikut :

```
Selamat Datang di UIN SUSKA RIAU
Press any key to continue . . .
```

Berikut akan dijelaskan tentang program diatas. Penjelasan disini masih bersifat global. Detil mengenai beberapa hal akan dijelaskan kemudian.

1. **public class Pertama** adalah nama kelas yang kita buat (dalam java setiap membuat program berarti membuat sebuah kelas).
2. **public static void main(String args[])** adalah permulaan fungsi utama dalam java. Kata kunci **void** didepan main merupakan keharusan pada java.
3. **System.out.println** memberikan hasil/output.

*Catatan :*
*Program java harus disimpan dengan nama class-nya. Huruf besar dan kecil dibedakan (case sensitif). Secara kesepakatan penulisan kelas sangat disarankan diawali dengan huruf kapital.*

## 3. LATIHAN

Buat program Java untuk memasukkan dan kemudian menampilkan data pribadi anda seperti berikut :

```
public class Dua
{
public static void main(String args[])
{
```



```
System.out.println("BIODATA PRIBADI");

System.out.println("==================
==========");
  System.out.println("Nama    : Elin
Haerani");
  System.out.println("NoMhS   :
961069");
  System.out.println("Jurusan: Teknik
Informatika");
}
}
```

Pada program di atas tambahkan data Jenis_Kelamin, Tgl_Lahir, Alamat.

## 4. TUGAS

1. Buatlah program untuk menampilkan 5 jurusan yang ada di FST UIN SUSKA RIAU, output yang diinginkan sebagai berikut :

```
FST UIN SUSKA RIAU
======================
Program Studi
1. Teknik Informatika/S1
2. Sistem Informasi/S1
3. Teknik Industri/S1
4. Teknik Elektro/S1
5. Matematika Terapan/S1

UIN SUSKA RIAU...TERDEPAN
```

2. Tugas dari dosen pengampu

# BAB 3
# OPERATOR

## 1. TEORI SINGKAT

Operator adalah simbol khusus yang menyajikan operasi khusus pada satu, dua, atau tiga operand dan kemudian mengembalikan hasilnya. Operator-operator tersebut digunakan untuk membentuk ekspresi. Sebuah konstanta atau entitas dengan sendirinya merupakan ekspresi, demikian juga kombinasi dari konstanta dan variabel dengan operator. Suatu ekspresi yang diikuti dengan sebuah titik koma adalah sebuah pernyataan. Operator mempunayi sifat- sifat

| Sifat | Keterangan | Contoh |
|---|---|---|
| Unary | Operator yang hanya melibatkan 1 operand | -1 |
| Binary | Operator yang hanya melibatkan 2 operand | 1 + 2 |
| Tenary | Operator yang hanya melibatkan 3 operand | 1 + 2 * 2 |



Pada saat kita akan menggunakan operator-operator dari bahasa pemrograman Java, kita harus mengetahui terlebih dahulu operator yang mana yang mempunyai *presedence* yang lebih tinggi. Operator di dalam tabel 2.1. dituliskan sesuai dengan *presedence* ordernya. Semakin ke bawah, maka *presedence*-nya lebih rendah. Operator dengan *presedence* yang lebih tinggi dikerjakan lebih dulu dari pada operator dengan *presedence* yang lebih rendah. Operator yang ada di dalam baris yang sama mempunyai presedence yang sama. Pada saat operator dari *presedence* yang sama muncul di dalam ekspresi yang sama, harus diatur yang mana yang harus dikerjakan lebih dulu. Semua operator biner kecuali untuk operator pemberian dikerjakan dari kiri ke kanan. Operator pemberian dikerjakan dari kanan ke kiri.

**Tabel 2.1. Precedence Operator**

| Operator | **Precedence** |
|---|---|
| *Postfix* | *expr++ expr--* |
| *Unary* | *++expr --expr +expr -expr* ~ ! |
| Multiplikasi | * / % |
| Aditif | + - |
| Pergeseran | << >> >>> |
| Relasional | < > <= >= instanceof |
| Persamaan | == != |
| bitwise AND | & |
| bitwise exclusive OR | ^ |
| bitwise inclusive OR | \| |
| Logika AND | && |
| Logika OR | \|\| |
| *Ternary* | ? : |
| Pemberian | = += -= *= /= %= &= ^= \|= <<= >>= >>>= |



**Operator Aritmatika**

| Operator | Keterangan |
|---|---|
| + | Penjumlahan (tanda plus) |
| - | Pengurangan (tanda minus) |
| * | Perkalian |
| / | Pembagian |
| % | Sisa Pembagian |

Operator *, / dan % mempunyai prioritas yang sama, tetapi lebih tinggi daripada + atau -.

## 2. CONTOH LATIHAN

Buatlah program seperti berikut :

```
import java.util.Scanner;
public class Jumlah
{
   public static void main(String
args[])
   {
      Scanner masuk = new
Scanner(System.in);
      int A,B, jumlah;
      System.out.print("Bilangan 1 :
");
      A = masuk.nextInt();
      System.out.print("Bilangan 2 :
");
      B = masuk.nextInt();
```

```
      jumlah = A + B;
      System.out.println("Jumlah = " +
jumlah);
   }
}
```

```
Hasil Output :

Bilangan 1 : 2
Bilangan 2 : 4
Jumlah = 6
Press any key to continue . . .
```

## 3. LATIHAN

Buat program dengan menggunakan operator aritmatika, seperti berikut ini :

```
public class Aritmatika
{
   public static void main(String
args[])
   {
   System.out.println("1 + 2 = " + (1
+ 2));
   System.out.println("3 * 2 = " + (3
* 2));
   System.out.println("7 / 3 = " + (7
/ 3));
   System.out.println("3 - 2 = " + (3
- 2));
   System.out.println("13% 5 = " + (13
% 5));
```



```
   System.out.println("");
   System.out.println("1 + 2 * 3 = " +
(1 + 2 * 3));
   System.out.println("1 + 2 / 2 = " +
(1 + 2 / 2));
   System.out.println("9 - 2 % 2 = " +
(9 - 2 % 2));
   }
}
```

```
Hasil Output :

1 + 2 = 3
3 * 2 = 6
7 / 3 = 2
3 - 2 = 1
13% 5 = 3

1 + 2 * 3 = 7
1 + 2 / 2 = 2
9 - 2 % 2 = 9
Press any key to continue . . .
```

## 4. TUGAS

1. Dengan menggunkan operator aritmatika buatlah untuk menjumlahkan, mengalikan, membagi dan sisa pembagian dari 2 bilangan yang diinpukan dengan keyboard, output yang diingikan sebagai berikut :

```
Bilangan 1 : 7
Bilangan 2 : 2

Hasil Operator Aritmatika
=========================
Jumlah = 9
Kurang = 5
Kali   = 14
Bagi   = 3
Sisa   = 1
Press any key to continue . . .
```

2. Buatlah program untuk menghitung keliling dan luas lingkaran
3. Tugas dari dosen pengampu



# BAB 4
# SEKUENSI

## 1. TEORI SINGKAT

Sekuensi adalah pemrograman sederhana yang hanya dapat memecahkan masalah-masalah yang sederhana. Masalah yang dapat diselesaikan dengan sekuensi saja biasanya hanya satu masalah kecil yang berdiri sendiri. Pembahasan sebelumnya, sejauh ini menggunakan prinsip sekuensi. Disamping itu, pada bagian ini kita juga akan mempelajari bagaimana membuat pseudocode sebelum implementasi ke bahasa pemrograman.

## 2. CONTOH LATIHAN

Buatkan program untuk menghitung konversi dari meter ke centimeter dan dari inci ke centimeter seperti berikut ini :

1 meter = 100

1 inci = 2.54 cm

```
import java.util.Scanner;
public class Konversi
{
    public static void main(String
args[])
      {
      Scanner masuk = new
Scanner(System.in);
      float m, cm, inci;
      System.out.print("Masukan ukuran
dalam Meter:");
      m = masuk.nextFloat();
      cm = m * 100;
      inci = m * 100 / 2.54f;
      System.out.println("Ukuran dalam
CM = " + cm);
      System.out.println("Ukuran dalam
Inci = " + inci);
      }
}
```

**Hasil Output**

```
Masukan ukuran dalam Meter :
50
Ukuran dalam CM = 5000.0
Ukuran dalam Inci = 1968.5039
```



## 3. LATIHAN

a. Program untuk konnversi suhu dari Celcius ke Fahreinhet.

Rumus

Fahreinhet = 9/5 * celcius + 32

```
import java.util.Scanner;
public class KonversiSuhu
{
     public static void main(String
args[])
     {
     Scanner masuk = new
Scanner(System.in);
     int celcius;
     float fahreinhet;
     System.out.print("Masukkan
derajat Celcius : ");
     celcius=masuk.nextInt();
     fahreinhet = 9f/5f * celcius +
32f;
     System.out.println("Fahreinhet :
"+fahreinhet);
     }
}
```

**Hasil Output**

```
Masukkan derajat Celcius : 30
Fahreinhet : 86.0
```

b. Tambahkan program diatas untuk menghitung konversi dari celcius ke Reamur dan Kelvin

Reamur = 4/5 * Celcius

Kelvin = Celcius + 273.

## 4. TUGAS

1) Buatlah program untuk mencari volume dari sebuah tabung dengan inputan jari – jari dan tinggi tabung.
2) Ditambah dengan tugas dari dosen pengampu



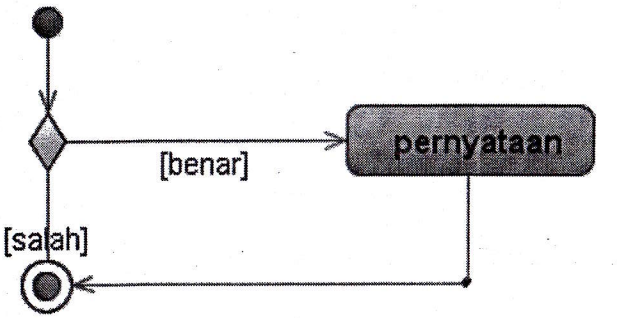
# BAB 5
# PERNYATAAN IF

## 1. TEORI SINGKAT

Pernyataan seleksi dengan IF akan mempunyai beberapa bentuk. Bentuk yang pertama adalah IF dengan satu pilihan. Bentuk umumnya adalah sebagai berikut.

```
if (kondisi)
        pernyataan;
```

Activity diagramnya adalah seperti gambar 3.1.




Gambar 5.1. Activity Diagram untuk pernyataan IF

Keterangan :

- Kondisi digunakan untuk menentukan pengambilan keputusan. Jika kondisi bernilai benar, maka pernyataan dikerjakan
- Pernyataan, berisi perintah-perintah dan akan dijalankan jika kondisi bernilai benar. Pernyataan disini bisa berupa pernyataan tunggal maupun majemuk.

## 2. CONTOH LATIHAN

Dengan menggunakan TextPad ketikkan program program – program berikut

**Program 1**

```
import java.util.Scanner;
public class IfSatuPilihan
{
public static void main(String args[])
{
Scanner masuk = new
Scanner(System.in);
int bil;
System.out.print("Masukkan bilangan :
");
bil=masuk.nextInt();
if (bil==0)
System.out.println("Bilangan Nol");
}}
```



**Hasil Output**

```
Masukkan bilangan : 0
Bilangan Nol
```

masukkan angka 7 dan apa hasilnya?

Modifikasi program diatas sehingga menjadi seperti berikut :

```
import java.util.Scanner;
public class IfDuaPilihan
{
     public static void main(String
args[])
     {
          Scanner masuk = new
Scanner(System.in);
          int bil;
          System.out.print("Masukkan
bilangan : ");
          bil=masuk.nextInt();
          if (bil==0)

     System.out.println("Bilangan
Nol");
          else
          System.out.println("Bilangan
Bukan Nol");
     }
}
```

masukkan angka 7 dan jelaskan hasilnya!

## 3. LATIHAN

**Program untuk memilih jurusan**

```
import java.util.Scanner;
public class IfJurusan
{
     public static void main(String
args[])
     {
          Scanner masuk = new
Scanner(System.in);
          int pil;
          System.out.print("Masukkan
pilihan : ");
          pil = masuk.nextInt();
          if (pil==1)
           System.out.println("Jurusan
MTK/S1");
           else if(pil==2)
           System.out.println("Jurusan
TE/S1");
           else if(pil==3)
           System.out.println("Jurusan
TIF/S1");
           else if(pil==4)
           System.out.println("Jurusan
TI/S1");
           else if(pil==5)
           System.out.println("Jurusan
SI/S1");
           else
           System.out.println("Pilihan
Salah!!!");}}
```



**Hasil output**

```
Masukkan pilihan : 4
Jurusan TI/S1
Press any key to continue . . .
```

## 4. TUGAS

Buat program untuk mendapatkan nilai determinan dari persamaan kuadrat. Kemudian akan ditampilkan pernyataan sesuai hasil determinannya.

Rumus: D=b*b-(4*a*c)

D=0 → akar kembar

D>0 → akar beda

D<0 → akar imajiner

Masukkanya adalah nilai a,b,c

Contoh output :

```
masukkan nilai a:2
masukkan nilai b:4
masukkan nilai c:2
akar kembar
```

# BAB 6
# PERNYATAAN SWITCH

## 1. TEORI SINGKAT

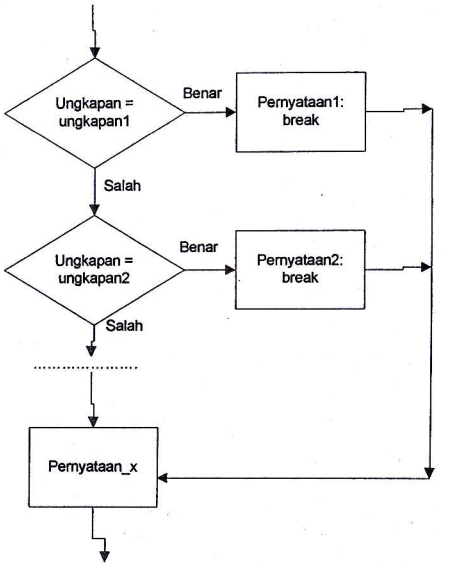
Pernyataan switch adalah pernyataan yang digunakan untuk menjalankann salah satu pernyataan dari beberapa kemungkinan pernyataan, berdasarkan nilai dari sebuah ungkapan dan nilai penyeleksi. Setiap ungkapan diungkapkan dengan sebuah nilai integral konstan, seperti sebuah nilai dengan tipe byte, short, int atau char.

Bentuknya :

```
switch (ungkapan)
{
    case ungkapan1:
        pernyataan1;
        break;
case ungkapan2:
        pernyataan2;
        break;
        ................
        default:
```



```
        pernyataan_x;
}
```




Keterangan :

- ungkapan1, ungkapan2 dan seterusnya dilakukan secara berurutan dimulai dari yang pertama, sekiranyanya cocok pernyataan yang mengikuti **case** dijalankan.
- **break** ditemukan dari eksekusi pernyataan **switch** berakhir
- **default** hanya akan dijalankan jika ungkapan pada bagian case tidak ada yang cocok.

## 2. CONTOH LATIHAN

Buatlah program berikut ini :

```
import java.util.Scanner;
public class CaseJurusan
{
   public static void main(String
args[])
   {
      Scanner masuk = new
Scanner(System.in);
      int pil;
      System.out.print("Masukkan
pilihan : ");
      pil = masuk.nextInt();
      switch (pil) {
       case
1:System.out.println("Jurusan
MT/D3");break;
       case
2:System.out.println("Jurusan
TE/D3");break;
       case
3:System.out.println("Jurusan
TIF/D3");break;
       case
4:System.out.println("Jurusan
TI/S1");break;
       case
5:System.out.println("Jurusan
SI/S1");break;
```



```
       default:
   System.out.println("Pilihan
Salah!!!");
                 break;
      }
   }
}
Hasil Output :

Masukkan pilihan : 2
Jurusan TK/D3
Press any key to continue . . .
```

Uji program di atas dengan memasukan angka 4 dan 6

## 3. LATIHAN

Buat program dengan menggunakan pernyataan switch, seperti berikut ini :

```
import java.util.Scanner;
public class CaseTV
{
   public static void main(String
args[])
   {
      Scanner masuk = new
Scanner(System.in);
      int pil;
      System.out.print("Masukkan
pilihan : ");
```

```
      pil = masuk.nextInt();
      switch (pil) {
       case
1:System.out.println("RCTI");break;
      case
2:System.out.println("SCTV");break;
      case
3:System.out.println("TPI");break;
      case
4:System.out.println("INDOSIAR");break
;
      case
5:System.out.println("TRANS7");break;
      case
6:System.out.println("TVONE");break;
      case
7:System.out.println("METRO");break;
      case
8:System.out.println("GLOBAL");break;
      case
9:System.out.println("TRANS7");break;
      case
10:System.out.println("TVRI");break;
      default:
   System.out.println("Salah masukkan
pilihan");
break;
      }
   }
}
```

Jelaskan hasil output yang diadapat dari pilihan yang anda lakukan.



## 4. TUGAS

1. Buatlah program dengan menggunakan pernyataan switch untuk memilih kendaraan yang akan dipakai ketika berpergian, sbb:
   - Pilihan 1 Naik Pesawat Terbang
   - Pilihan 2 Naik Kereta Api
   - Pilihan 3 Naik Bus
   - Pilihan 4 Naik Taksi
   - Pilihan 5 Naik Mobil Pribadi
   - Pilihan 6 Naik Motor
   - Jika tidak ada diantara pilihan 1 – 6, beri komentar "Anda salah Memilih"
2. Setelah anda melakukan praktikum dengan menggunakan pernyataan if dan switch jelaskan kekurangan dan kelebihan dari kedua pernytaan tersebut

3. Tugas dari dosen pengampu



# BAB 7
# PERULANGAN DENGAN WHILE

## 1. TEORI SINGKAT

Pernyataan ini berguna untuk memproses suatu pernyataan atau beberapa pernyataan beberapa kali. Selama ungkapan bernilai benar, pernyataan akan selalu dikerjakan.

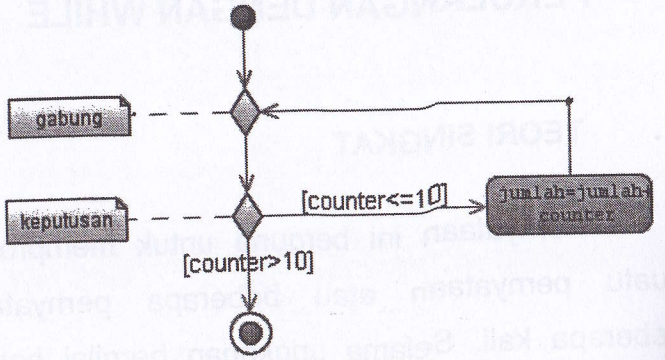
Bentuknya :

```
while (ungkapan)
        Pernyataan;
```

Keterangan :

- bagian pernyataan akan diekseskusi selama ungkapan dalam **while** bernilai benar.
- Pengujian terhadap ungkapan pada **while** dilakukan sebelum bagian pernyataan.
- Kemungkinan pernyataan pada **while** tidak dijalankan sama sekali, jika ketemu kondisi yang pertama kali bernilai salah.

Activity diagramnya adalah seperti gambar berikut :




Catatan :

Pernyataan perulangan dengan while akan selalu dikerjakan jika ungkapan selalu benar. Oleh karena itu, kita harus membuat kondisi suatu saat ungkapan bernilai salah agar perulangan berakhir.

## 2. CONTOH LATIHAN

Buatlah program seperti berikut ini :

```
import java.util.Scanner;
  public class UlangWhile1
  {
```



```
    public static void main(String
args[])
    {
        Scanner masuk = new
Scanner(System.in);
        int bil;
        bil=1;
        while (bil<=5) {

  System.out.println(bil);
            bil++;
        }
    }
  }
```

```
Hasil Output
1
2
3
4
5
Press any key to continue...
```

Ubah pernyataan **bil=1** menjadi **bil=5**, pernyataan while **(bil<=5)** dengan while**(bil>=1)** dan **bil++** menjadi **bil--**, amati hasil outputnya.

## 3. LATIHAN

Dengan while, buatlah program untuk mencetak bilangan genap dari 0 sampai dengan 10.

```
import java.util.Scanner;
  public class UlangWhile3
  {
    public static void main(String
args[])
    {
        Scanner masuk = new
Scanner(System.in);
        int bil;
        bil=2;
        while (bil<=10) {
          System.out.println(bil);
          bil+=2;
        }
    }
  }
Hasil Output :
2
4
6
8
10
Press any key to continue . . .
```

Ubah program di atas untuk menampilkan bilangan ganjil saja yaitu 1, 3, 5, 7, 9



## 4. TUGAS

1. Hitunglah rata-rata bilangan positif, dimana banyaknya data ditentukan dari data yang dimasukan dengan keyboard, hasil output yang diinginkan sbb:

```
Banyaknya data : 4
Data ke-1 : 3
Data ke-2 : 5
Data ke-3 : 2
Data ke-4 : 6
Rata-rata : 4.0
Jumlah    : 16.0
Press any key to continue . . .
```

2. Tugas dari dosen pengampu

# BAB 8
# PERULANGAN DENGAN DO WHILE

## 1. TEORI SINGKAT

Seperti halnya perulangan dengan while, perulangan dengan do ... while ini juga digunakan untuk mengerjakan sebuah atau sekelompok pernyataan berulang-ulang. Bedanya dengan while adalah pernyataan do ... while akan mengecek kondisi di belakang, sementara while cek kondisi ada di depan.

Bentuknya :

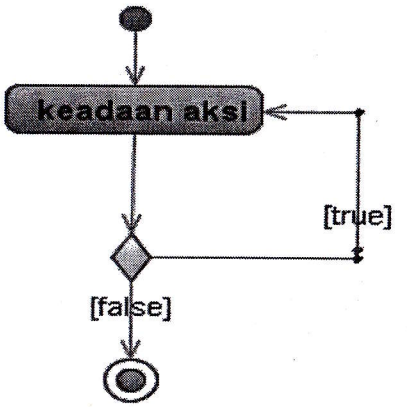
```
do
{
        pernyataan1;
        pernyataan2;
        .....
        pernyataan_N;
}
while (ungkapan)
```



Keterangan :

- Bagian pernyataan1 hingga pernyataanN dijalankan secara berulang sampai ungkapan bernilai salah.
- Pengujian ungkapan dilakukan setelah bagian pernyataan, maka pada pernyataan **do ... while** minimal akan dijalankan sekali, karena begitu masuk ke blok perulangan, tidak ada cek kondisi tetapi langsung mengerjakan pernyataan.

Activity diagramnya :



## 2. CONTOH LATIHAN

Tuliskan program berikut ini :

```
import java.util.Scanner;
  public class UlangDo1
  {
    public static void main(String
args[])
    {
          Scanner masuk = new
Scanner(System.in);
          int bil;
          bil=1;
          while (bil<=10) {
            System.out.println(bil);
            bil+=2;
         }
    }
  }
Hasil Output :
1
3
5
7
9
Press any key to continue . . .
```

Ubah program di atas agar mendapat hasil output yang bilangan genap saja (0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20)



## 3. LATIHAN

Buatlah program mencetak konversi suhu dari celcius ke fahreinhet mulai dari 1 sampai 10 dengan membuat tabel.

```
public class UlangDo2
{
  public static void main(String
args[])
  {
    int c;
    double f;
    System.out.println("----------
----------");
    System.out.println("CELCIUS
FAHREINHEIT");
    System.out.println("----------
----------");
    c=1;
    do
{
    f=1.8 * c + 32;

  System.out.println("Celcius:"+c+"Fa
hreinhet: "+f);
    c++;
    } while (c<=10);
    System.out.println("----------
----------");
  }
}
```

Hasil Output :

```
-----------------------------
 CELCIUS          FAHREINHEIT
-----------------------------
Celcius : 1 Fahreinhet : 33.8
Celcius : 2 Fahreinhet : 35.6
Celcius : 3 Fahreinhet : 37.4
Celcius : 4 Fahreinhet : 39.2
Celcius : 5 Fahreinhet : 41.0
Celcius : 6 Fahreinhet : 42.8
Celcius : 7 Fahreinhet : 44.6
Celcius : 8 Fahreinhet : 46.4
Celcius : 9 Fahreinhet : 48.2
Celcius : 10 Fahreinhet : 50.0
-----------------------------
Press any key to continue . . .
```

## 4. TUGAS

1. Hitunglah rata-rata bilangan positif, dimana banyaknya data ditentukan dari data yang dimasukan.
2. Tugas dari dosen pengampu



# BAB 9

# PERULANGAN DENGAN FOR

## 1. TEORI SINGKAT

Sama seperti pernyataan perulangan while dan do...while, pernyataan for juga digunakan untuk mengerjakan pernyataan atau sekelompok pernyataan secara berulang. Bedanya adalah dengan pernyataan for perulangan akan dikerjakan dalam hitungan yang sudah pasti, sementara while dan do...while tidak.

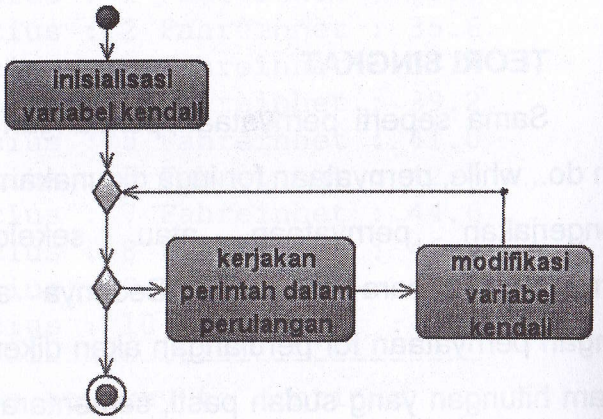
Bentuknya :

```
for (ungkapan1;ungkapan2;ungkapan3)
        Pernyataan;
```

Keterangan :

- ungkapan1 merupakan pernyataan inisialisasi
- ungkapan2 sebagai kondisi yang menentukan pengulangan terhadap pernyataan atau tidak

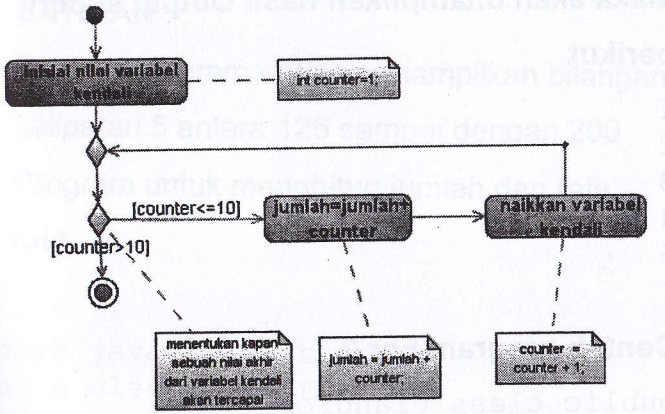
- ungkapan3 digunakan sebagai pengatur variabel yang digunakan didalam ungkapan1




Gambar 4.15. Activity Diagram untuk perulangan dengan FOR

Contoh activity diagram untuk perulangan dengan for.






Gambar 9.1. Activity Diagram untuk perulangan dengan FOR

## 2. CONTOH LATIHAN

Ketikkan program program – program berikut

**Contoh Program For 1**

```
public class UlangFor
{
    public static void main (String
args[])
    {
     int bil;
     for (bil=1;bil<=5;bil++)
        System.out.println(bil);
    }
}
```

**Maka akan ditampilkan hasil Output seperti berikut**

```
1
2
3
4
5
```

**Contoh program For 2**

```
public class UlangFor2
{
    public static void main (String
args[])
    {
      int bil;
     for (bil=5;bil>=1;bil--)
      System.out.println(bil);
    }
}
```

**OUTPUT:**

```
5
4
3
2
1
```



## 3. LATIHAN

- Buatlah program untuk menampilkan bilangan kelipatan 5 antara 125 sampai dengan 200
- Program untuk menghitug jumlah dan rata – rata

```
import java.util.Scanner;
public class UlangFor6
{
    public static void main(String
args[])
    {
        Scanner masuk = new
Scanner(System.in);
        int i;
        float n, jum, x, rata;
        System.out.print("Banyaknya
data : ");
        n = masuk.nextFloat();
        jum=0;
        for (i=1;i<=n;i++){
          System.out.print("Data ke-
"+i+" : ");
          x=masuk.nextFloat();
          jum += x;
        }
        System.out.println("Jumlah :
"+jum);
    }}
```

**Hasil output**

```
Banyaknya data : 3
Data ke-1 : 21
Data ke-2 : 7
Data ke-3 : 14
Jumlah : 42.0
```

c. Modifikasi program diatas untuk menghitung rata – rata dari sejumlah data yang dimasukkan!

**4. TUGAS**

1. Buat sebuah program java yang digunakan menjumlahkan bilangan bulat positif yang lebih kecil dari 100.
2. Buatlah program untuk menampilkan bilangan ganjil yang batas awal dan akhirnya dimasukkan melalui keyboard.



# BAB 10
# STRING

## 1. TEORI SINGKAT

String adalah klas yang menangani deretan karakter. Klas ini mendukung sejumlah metode yang sangat berguna untuk memanipulasi string, misalnya untuk mengkonversikan setiap huruf kecil menjadi huruf besar atau sebaliknya, memperoleh jumlah karakter dan sebagainya.

Klas string memiliki banyak konstruktor, seperti tabel berikut :

| Konstruktor | Keterangan |
|---|---|
| String() | Menciptakan obyek string yg berisi string kosong (jumlah karakter = 0) |
| String(char[]v) | Menciptakan obyek string yg berisi string yg berasal dari array yg dirujuk oleh v |

| | |
|---|---|
| String(String v) | Menciptakan obyek string yg isinya sama dengan obyek string argumennya |

Metode dalam klas string memperlihatkan sejumlah metode penting dalam klas string, seperti :

- copyValueOf(char data[])
- copyValueOf(char data[], int offset, int jum)
- valueOf(boolean b)
- valueOf(double c)
- cocat(String s)
- length()
- trim()
- dan lain-lain

Klas StringBuffer adalah klas yg menyimpan string yang konstan, begitu obyek string telah diciptakan maka string tidak dapat diubah. Konstruktor klas ini antara lain :

- StringBuffer() digunakan untuk menciptakan StringBuffer yang kosong



- StringBuffer(int n) digunakan untuk menciptakan StringBuffer dengan n karakter
- StringBuffer(String s) digunakan untuk menciptakan StringBuffer dengan string berupa s

## 2. CONTOH LATIHAN

Tuliskan program berikut ini :

```
public class ContohString
{
public static void main(String
args[])
{
   byte data[] = new byte[6];
   data[0] = 64;
   data[1] = 65;
   data[2] = 66;
   data[3] = 67;
   data[4] = 68;
   data[5] = 69;
   String s1 = "Selamat Pagi";
   String s2 = new String("Good
Morning");
   String s3 = new String(data);
   String s4 = new String(data, 2,
3);
   System.out.println("s1 = " + s1);
   System.out.println("s2 = " + s2);
   System.out.println("s3 = " + s3);
   System.out.println("s4 = " + s4);
```

```
    }
}
```

**Hasil output :**

```
s1 = Selamat Pagi
s2 = Good Morning
s3 = @ABCDE
s4 = BCD
Press any key to continue . . .
```

Pada program di atas, pernyataan seperti :

```
String s1 = "Selamat Pagi";
```

Sebenarnya identik dengan :

```
String s1 = new String("Selamat Pagi");
```

Pernyataan

```
String s3 = new String(data);
```

akan membuat string yang tersusun atas karakter-karakter yang nilainya sama seperti elemen-elemen pada array data, maka s3 berisi string @ABCDE adalah karakter @ = 64, A=65 dan seterusnya.

Pernyataan :

```
String s4 = new String(data, 2, 3);
```



Angka 3 menyatakan jumlah karakter yg menyusun string dan angka 2 menyatakan karakter pertama pada string, hasil diambil pd indeks ke-2 array.

## 3. LATIHAN

Buatlah program untuk mengubah huruf kecil menjadi huruf besar dan sebaliknya dan juga untuk menghitung jumlah karakter, seperti berikut ini :

```
public class ContohString1
{
   public static void main(String args[])
   {
   String st ="Tes, Tes, tes... 123";
   System.out.println("toLowerCase : " +
st.toLowerCase());
   System.out.println("toUpperCase : " +
st.toUpperCase());
   System.out.println("Dgn Trim     : " +
'[' + st.trim() + ']' );

   System.out.println("Jumlah karakter = "
+ st.length());
   }
}
```

**Hasil Output :**

```
toLowerCase : tes, tes, tes... 123
toUpperCase : TES, TES, TES... 123
Dgn Trim    : [Tes, Tes, tes... 123]
```

```
Jumlah karakter = 20
Press any key to continue . . .
```

Buatlah program menggunakan StringInsert berikut ini :

```
public class StringInsert
{
     public static void main(String
args[])
     {
     StringBuffer sbuf = new
StringBuffer("THE TIF");
     sbuf.insert(4, "REAL ");
     System.out.println("Hasil : ");
     System.out.println(sbuf.toString(
));

     }
}
Hasil Output :
Hasil :
THE REAL TIF
Press any key to continue . . .
```

## 4. TUGAS

Jika ada program dengan menggunakan StringBuffer sebagai berikut :

```
public class StrBuf
{
   public static void main(String args[])
```



```
   {
   StringBuffer sbuf = new
StringBuffer(25);
   System.out.println("Isi : " +
sbuf.toString());
   System.out.println("Kapasitas : " +
sbuf.capacity());
   System.out.println("Panjang   : " +
sbuf.length());
   sbuf.append("Selamat");
   sbuf.append("Belajar Java");

   }
}
Hasil output :
Isi :
Kapasitas : 25
Panjang   : 0
Press any key to continue . . .
```

Tambahkan program tersebut di atas agar menghasilkan output sebagai berikut :

```
Isi :
Kapasitas : 25
Panjang   : 0

Isi : Selamat Belajar Java
Kapasitas : 25
Panjang   : 20

Isi : Selamat
Kapasitas : 25
```

```
Panjang    : 7
Press any key to continue . . .
```



# BAB 11
# METHOD TANPA PARAMETER

## 1. TEORI SINGKAT

Method (atau dalam beberapa bahasa pemrograman sering disebut fungsi atau prosedur) adalah sub program yang membiarkan seorang programer untuk membagi program dengan membagi masalah ke dalam beberapa sub masalah yang bisa diselesaikan secara modular. Dengan cara demikian, maka pembuatan program bisa lebih dimanajemen.

Kelas (*class*) adalah program java yang akan dieksekusi. Method ada di dalam kelas. Java mempunyai kumpulan kelas yang sudah dimiliki yang tersimpan di dalam paket-paket. Kumpulan kelas tersebut ada di dalam *Java Application Interface* (Java API) atau *Java class libraries* dan beberapa *libraries* lainnya.

**FORMAT METHOD SECARA UMUM**

```
tipe_return-value
nama_method(parameter1, parameter2, ...,
parameterN)
{
    deklarasi dan pernyataan;
}
```

Elemen yang diperlukan dari deklarasi method adalah tipe kembalian method, nama, kurung buka dan tutup ( ) dan isi method yang diawali dan diakhiri dengan kurung kurawal buka dan tutup { }. Secara umum, deklarasi method mempunyai 6 komponen, yaitu

1. Modifier - seperti public, private, dan yang lain yang akan kita pelajari kemudian.
2. Tipe kembalian (*return type*)—tipe data dari nilai yang dikembalikan oleh method, atau void jika method tidak mempunyai nilai kembalian.
3. Nama method—aturan untuk penamaan field diterapkan untuk nama method tetapi kesepakatannya adalah sedikit berbeda.



4. Daftar parameter – pemisah antar parameter input adalah koma, diawali oleh tipe datanya, yang diletakkan diantara tkita kurung ( ...daftar parameter.... ). Jika tidak ada parameter, harus menggunakan kurung buka tutup saja ( ).
5. Daftar exception—tidak akan masuk dalam pembahasan di sini
6. Isi method, diletakkan di antara kurung kurawal buka dan tutup { }—kode-kode method, termasuk deklarasi variabel lokal ada di sini.

## 2. CONTOH LATIHAN

Untuk membuat sebuah method, yang pertama kali perlu diperhatikan adalah nama method mempunyai aturan yang sama dengan penamaan variabel.

Contoh

Fungsi untuk menampilkan tulisan saja.

```
public class Fungsi1{

    public static void garis(){

    System.out.println("==========
===");
```

```
    }

    public static void main(String
args[]){
        garis();
    }
}
```

Keluaran dari program tersebut adalah

==========

Method diatas bersifat static dan bertipe void. Method yang bersifat static bisa langsung dipanggil dengan nama methodnya saja.

Sebuah method juga bisa dipanggil lebih dari satu kali.

## 3. LATIHAN

Tuliskan program berikut :

Program Latihan 1

```
public class Fungsi2
{
    public static void kalimat()
{
    System.out.println("Di dalam
method kalimat");
    }
```



```
    public static void main(String
args[])
{
        kalimat();
        System.out.println("Di dalam
main");
        kalimat();
    }
}
```

**Hasil output**

```
Di dalam method kalimat
Di dalam main
Di dalam method kalimat
```

Program Latihan 2

```
public class Fungsi3
{
    public static int jumlah(){
        int a = 7, b = 15;
        return (a + b);
    }

    public static void main(String
args[]){
System.out.println("Hasil pemanggilan
method jumlah");

    System.out.println(jumlah());
    }
}
```

# BAB 12

# METHOD DENGAN PARAMETER

## 1. TEORI SINGKAT

Method (atau dalam beberapa bahasa pemrograman sering disebut fungsi atau prosedur) adalah sub program yang membiarkan seorang programer untuk membagi program dengan membagi masalah ke dalam beberapa sub masalah yang bisa diselesaikan secara modular. Dengan cara demikian, maka pembuatan program bisa lebih dimanajemen.

Contoh :

```
1.  public class FungsiParameter
2.  {
3.  public static int jumlah(int a){
4.  return a;
5.  }

6.  public static void main(String
args[]){
7.  System.out.println("Hasil
pemanggilan methode ");
8.  System.out.println(jumlah(5));
```



```
9.  }
10. }
```

**Hasil Output**

```
Hasil pemanggilan method jumlah
5
Press any key to continue . . .
```

Parameter pada baris kedua disebut sebagai parameter formal, dan pada baris ke 8 disebut parameter aktual.

Ada 2 buah parameter yaitu

- parameter formal adalah parameter yang tertulis dalam definisi method
- Parameter aktual parameter yang berada pada inputan langsung pada saat penggunaan method tersebut.

Parameter bisa lebih dari satu dengan dipisahkan tanda koma,. Yang perlu diperhatikan pada saat pemanggilan method adalah jumlah, urutan dan

tipe parameter aktual harus sesuai dengan jumlah urutan dan tipe parameter formal.

Pemberian Variabel Dalam Method

Ada dua tipe data variable passing pada method, yaitu *pass-by-value* dan *pass-by-reference.*

***Pass-by-value***

Ketika *pass-by-value* terjadi, method membuat sebuah salinan dari nilai variable yang dikirimkan ke mthod. Walaupun demikian method tidak dapat secara langsung memodifikasi nilai variable pengirimnya meskipun parameter salinannya sudah dimodifikasi nilainya di dalam method.

***Pass-by-reference***

Ketika sebuah *pass-by-reference* terjadi, alamat memori dari nilai pada sebuah variable dilewatkan pada saat pemanggilan method. Ini tidak seperti pada *pass-by-value*, method dapat memodifikasi variable asli dengan menggunakan alamat memori



tersebut, meskipun berbeda nama variable yang digunakan dalam method dengan variable aslinya, kedua variable ini menunjukkan lokasi dari data yang sama.

## 2. CONTOH LATIHAN

Tuliskan program berikut :

```
public class Fungsi4a{

public static int jumlah(int a){
return (a + a);
}

public static void main(String
args[]){
System.out.println("Panggil      method
jumlah dengan parameter 5");
System.out.println(jumlah(5));
System.out.println("Panggil      method
jumlah dengan parameter 15");
System.out.println(jumlah(15));
}
}
```

**Hasil Output**

```
Panggil method jumlah dengan parameter
5
10
```

```
Panggil method jumlah dengan parameter
15
30
Press any key to continue . . .
```

**Program dengan pass-by-value**

```
public class TestPassByValue
{
public static void main(String [] args
)
{
int i =10;
System.out.println(i);
test(i);
System.out.println(i);
}
public static void test(int j)
{
j=33;
}
}
```

**Hasil Output**

```
10
10
Press any key to continue . . .
```

Pada program diatas kita memanggil method tes dan melewatkan nilai variable i sebagai parameter.



Nilai pada i disalinkan ke variable j pada method. Pada kondisi ini variable j adalah merupakan variable pengganti pada method test, jika nilai j berubah maka nilai pada variable I yang terletak pada main tidak akan ikut berubah walaupun awalnya variable j merupakan salinan dari variable i.

**Program dengan pass-by-reference**

```
class TestPassByReference
{
public static void main( String []
args)
{
//membuat array integer
int [] ages ={10,11,12};

//mencetak nilai array
for (int i=0; i<ages.length; i++)
{
System.out.println(ages[i]);
}

test(ages);
for (int i=0; i<ages.length; i++)
{
System.out.println(ages[i]);
}
```

```
}
public static void test(int[] arr)
{
//merubah nilai array
for (int i=0; i<arr.length; i++)
{
arr[i] = i+50;
}
}
}
```

**Hasil output**

```
10
11
12
50
51
52
Press any key to continue . . .
```

## 3. LATIHAN

Kerjakan Program berikut :

```
public class Fungsi4c
{
    public float jumlah(int a, float
b) //tanpa static
    {

       return (a+b);
    }
```



```
    public static void main(String
args[])
    {
        Fungsi4 obyek=new Fungsi4();
        System.out.print("panggil
method jumlah dengan parameter 5 dan
1.5, hasilnya = ");

System.out.println(obyek.jumlah(5,
1.5f));
        System.out.print("panggil
method jumlah dengan parameter 10 dan
2.2, hasilnya = ");

System.out.println(obyek.jumlah(10,
2.2f));
    }
}
```

**Hasil output**

```
panggil method jumlah dengan parameter
5 dan 1.5, hasilnya = 6.5
panggil method jumlah dengan parameter
10 dan 2.2, hasilnya = 12.2
```

Modifikasi program diatas dengan menambahkan satu parameter lagi.

## 4. TUGAS

1. Buat sebuah method yang digunakan untuk menghasilkan nilai paling kecil dari 3 bilangan yang dimasukkan sebagai parameter.
2. Tugas dari dosen pengampu



# BAB 13
# ARRAY / LARIK

## 1. TEORI SINGKAT

Larik adalah sebuah struktur data yang terdiri dari data yang bertipe sama. Ukuran larik bersifat tetap, larik akan mempunyai ukuran yang sama pada saat sekali dibuat. Larik dalam Java adalah obyek, disebut juga sebagai tipe referensi. Sedangkan elemen dalam larik Java bisa primitif atau referensi. Posisi dari larik biasa disebut sebagai elemen. Elemen larik dimulai dari 0 (nol). Penyebutan larik diberikan dengan cara menyebutkan nama lariknya dan diikuti dengan indeksnya, dimana indeks dituliskan diantara tanda kurung siku. Gambar 1. memperlihatkan gambaran larik dengan 10 elemen, dimana setiap elemennya bertipe integer, dengan nama A.

| Nama | A[0] | A[1] | A[2] | A[3] | A[4] | A[5] | A[6] | A[7] | A[8] | A[9] |
|---|---|---|---|---|---|---|---|---|---|---|
| Isi larik | 12 | -56 | 23 | 45 | -16 | -2 | 85 | 41 | 15 | 20 |

## 2.1. DEKLARASI DAN MENCIPTAKAN LARIK

Sebagai sebuah obyek, larik harus diciptakan dengan menggunakan kata cadang new. Deklarasi dan penciptaan variabel larik gambar 1 adalah sebagai berikut.

int A[] = new int[10];

larik dideklarasikan dan langsung diciptakan . Atau

int A[];

A = new int[10];

larik dideklarasikan, baru pada pernyataan berikutnya larik diciptakan.

## 2. CONTOH LATIHAN

Tuliskan program berikut :

```
import java.util.Scanner;
public class Larik1
{
```



```
    public static void main (String
args[])
    {
      Scanner masuk=new
Scanner(System.in);
      float nilai[]=new float[5];
      System.out.println("masukkan 5
buah data nilai");
        for(int i=0;i<5;i++)
        {
            System.out.print("Data
ke"+(i+1)+": ");

nilai[i]=masuk.nextFloat();
        }
      System.out.println("data nilai
yang dimasukkan");
        for(int i=0;i<5;i++)

System.out.println(nilai[i]);

    }
}
```

**Hasil Output**

```
masukkan 5 buah data nilai
Data ke1: 2
Data ke2: 4
Data ke3: 5
Data ke4: 7
Data ke5: 9
```

```
data nilai yang dimasukkan
2.0
4.0
5.0
7.0
9.0
Press any key to continue . . .
```

## 3. LATIHAN

Tuliskan program untuk menampilkan bilangan dari 1 sampai 10 dengan pangkatnya masing – masing berikut:

```
public class Larik4
{
public static void main(String args[])
{

int kuadrat[];
kuadrat = new int[10];
for (int i=0;i<10;i++)
{
kuadrat[i]=(i+1)*(i+1);
System.out.println("Kuadrat "+(i+1)+
                   " = "+kuadrat[i]);

}
}
}
```



**Hasil output**

```
Kuadrat 1 = 1
Kuadrat 2 = 4
Kuadrat 3 = 9
Kuadrat 4 = 16
Kuadrat 5 = 25
Kuadrat 6 = 36
Kuadrat 7 = 49
Kuadrat 8 = 64
Kuadrat 9 = 81
Kuadrat 10 = 100
```

Larik juga dapat diberikan nilai awal (diinialisasi) pada saat pendefinisian

```
public class Larik6
{
    public static void main(String
args[])
    {
    int hari []= {
    31, 28, 31, 30, 31, 30, 30, 31,
30, 31, 30, 31
    };

       for(int i = 0; i < 12; i++)
           System.out.println("Bulan
" + (i+1) + " = "
               +hari[i]);
    }
}
```

Hasil output :

```
Bulan 1 = 31
Bulan 2 = 28
Bulan 3 = 31
Bulan 4 = 30
Bulan 5 = 31
Bulan 6 = 30
Bulan 7 = 30
Bulan 8 = 31
Bulan 9 = 30
Bulan 10 = 31
Bulan 11 = 30
Bulan 12 = 31
```

## 4. TUGAS

Modifikasi program pada praktik Larik1 diatas untuk menghitung jumlah, rata – rata, serta nilai terbesar dan terkecilnya!



# BAB 14

# ARRAY / LARIK MULTI DIMENSI

## 1. TEORI SINGKAT

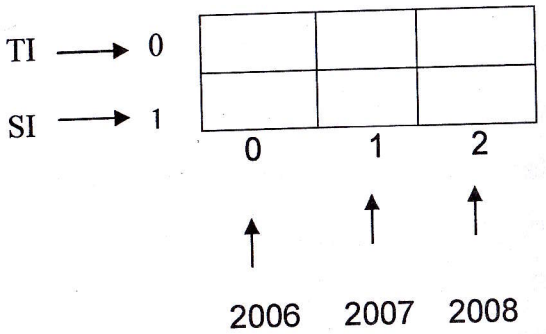
Kita juga bisa membuat variabel larik yang tipe elemennya adalah larik. Dengan cara demikian, kita membuat larik dua dimensi. Dengan larik dua dimensi, maka kita mempunyai elemen yang berindeks tidak hanya satu, tetapi dua. Kita bisa membayangkan larik dua dimensi tersebut seperti sebuah tabel yang berisi baris dan kolom. Penyebutan sel tabel selalu diikuti dengan penyebutan baris berapa dan kolom berapa.

Contoh :

Diberikan data kelulusan mahasiswa sebuah perguruan tinggi sebagai berikut.

| Jurusan | 2006 | 2007 | 2008 |
|---|---|---|---|
| Teknik Informatika | 110 | 125 | 135 |
| Sistem Informasi | 56 | 75 | 80 |

int data_lulus [2] [3]




| | | |
|---|---|---|
| [0] [0] | [0] [1] | [0] [2] |
| [1] [0] | [1] [1] | [1] [2] |

## 2. CONTOH LATIHAN

Tuliskan program berikut :

```
public class ArrayDimensiDua
{
     public static void main(String []
args)
     {
     int [][] piksel = new int[2][3];
     // mengisi elemen tertentu
     piksel[0][0] = 70;
     piksel[0][1] = 18;
     piksel[0][2] = 45;
```



```
     piksel[1][0] = 75;
     piksel[1][1] = 66;
     piksel[1][2] = 89;
     //menampilkan elemen array
     int i,j;
     for(i=0;i<2;i++){
          for (j=0; j<3;j++)

     System.out.print(piksel[i][j] +"
");
          System.out.println("");
     }
     }
}
```

Hasil Output

```
70 18 45
75 66 89
```

## 3. LATIHAN

Tuliskan program untuk menampilkan menampilkan matrik yang elemen – elemenya dimasukkan melalui keyboard :

## Program latihan 1

```
import java.util.Scanner;
public class ArrayDimensiDua2{
public static void main(String
args[]){
     Scanner masuk = new
Scanner(System.in);
     int nilai[][]= new int[2][3];
     System.out.println("Masukkan data
nilai");
     for (int i = 0; i < 2; i++)
     {
       for (int j = 0; j < 3; j++){
       System.out.print("("+ (i + 1
)+" , "+ (j+1) +")"+ " : ");
          nilai[i][j]=masuk.nextInt();
       }
     }

     System.out.println("Data nilai
yang dimasukan");
     for (int i = 0; i < 2; i++){
          for (int j = 0; j < 3; j++)

     System.out.print(nilai[i][j]+"
");
               System.out.println();}

}
}
```



## Program latihan 2

program untuk menjumlahkan matrik :

```
import java.util.Scanner;
public class JumlahMatriks{
          Scanner masuk = new
Scanner(System.in);
          public void
masukData(float data[][]){
               for (int i = 0; i < 3;
i++){
               for (int j = 0; j < 3;
j++){
System.out.print( "("+(i + 1 )+" , "+
(j+1)+")"
+ " : ");
          data[i][j]=masuk.nextFloat
();
                    }
               }
          }

          public float[][]
tambah(float AA[][],
          float BB[] []){
               float hasil[][]= new
float[3][3];
               for (int i = 0; i < 3;
i++)
                    for (int j = 0; j
< 3; j++)
                    hasil[i][j] =
AA[i][j] + BB[i][j];
```

```
                return hasil;
            }

public void tampilData(float data[][],
char nama){
                for (int i=0;i<3;i++){
                    for (int
j=0;j<3;j++)
            System.out.print(nama+"["+
(i + 1 )
            +"] ["+ (j+1) + "] = " +
data[i][j]+"   ");

            System.out.println();
                }
            }

            public static void
main(String args[]){
                float A[][] = new
float[3][3];
                float B[][] = new
float[3][3];
                float C[][] = new
float[3][3];
                JumlahMatriks jumlah =
new JumlahMatriks();
System.out.println("Masukkan data
matriks A");
                jumlah.masukData(A);
System.out.println("Masukkan data
matriks B");
                jumlah.masukData(B);
```



```
                C = jumlah.tambah(A,B);

            jumlah.tampilData(C,'C');
            }
}
```

## Hasil output

```
Masukkan data matriks A
(1 , 1) : 2
(1 , 2) : 3
(1 , 3) : 1
(2 , 1) : 4
(2 , 2) : 2
(2 , 3) : 4
(3 , 1) : 1
(3 , 2) : 2
(3 , 3) : 3


Masukkan data matriks B
(1 , 1) : 5
(1 , 2) : 4
(1 , 3) : 3
(2 , 1) : 6
(2 , 2) : 2
(2 , 3) : 3
(3 , 1) : 4
(3 , 2) : 1
(3 , 3) : 3
C[1] [1] = 7.0   C[1] [2] = 7.0   C[1]
[3] = 4.0
```

```
C[2] [1] = 10.0  C[2] [2] = 4.0  C[2]
[3] = 7.0
C[3] [1] = 5.0  C[3] [2] = 3.0  C[3]
[3] = 6.0
Press any key to continue . . .
```

## 4. TUGAS

Buat sebuah program dengan larik untuk menyimpan 10 data mahasiswa yaitu berupa NIM, nama dan jurusan



# BAB 15
# KELAS DAN OBYEK I

## 1. TEORI SINGKAT

Kelas dalam java bisa dianalogikan dalam kehidupan sehari-hari sebagai sebuah kelompok yang mempunyai sifat dan tingkah laku yang serupa atau secara umum serupa. Kelompok di sini dapat kelompok benda maupun makluk hidup. Misal orang, mobil, motor, sepeda, ayam, kucing, bunga.

### a. Mendeklarasikan sebuah kelas

Pada bagian ini, kita akan membahas kelas secara luas. Termasuk atribut-atribut dan segala yang berkaitan dengan kelas yang ada di dalamnya. Mari kita perhatikan definisi kelas berikut.

```
class KelasKu {
  //deklarasi field, konstruktor dan method
}
```

Itu adalah sebuah deklarasi kelas dengan nama KelasKu. Deklarasi ini sudah sering kita buat sebelumnya dengan menambah kata kunci public diawalnya. Isi dari kelas (daerah antara dua tkita kurung kurawal) berisi semua kode yang disediakan untuk obyek yang diciptakan dari kelas, yaitu konstruktor untuk inisialisasi obyek baru, deklarasi field yang menetapkan keadaan kelas dan obyeknya dan method untuk mengimplementasikan lingkungan dari kelas dan obyeknya.

Secara umum, deklarasi kelas dapat termasuk komponen-komponen

1. Modifier seperti *public, private* dan modifier yang lain yang akan kita bicarakan kemudian.
2. Nama kelas, dengan diawali huruf besar sebagai kesepakatan.
3. Nama dari induk kelasnya (*superclass*), jika ada, diawali dengan kata kunci extends. Sebuah kelas hanya boleh mempunyai satu induk
4. Daftar interface (dipisahkan dengan tkita koma) yang akan diimplementasikan dalam kelas, jika ada, diawali dengan kata kunci *implements*. Sebuah kelas boleh mengimplementasikanlebih dari satu *interface*
5. Isi dari kelas yang diawali dan diakhiri dengan tanda kurung kurawal buka dan tutup { }



Bagian yang tidak kalah pentingnya adalah deklarasi variabel anggota. Ada beberapa macam variabel yang ada di bagian ini.

- Variabel anggota dalam sebuah kelas – ini disebut *fields*. Fields ini terletak di luar method. Dan bisa diakses dai method dengan menggunakan referensi ke kelas yang memiliki field tersebut (dengan memperhatikan aturan akses modifier)
- Variabel dalam sebuah method atau blok kode – ini disebut variabel lokal. Variabel ini biasanya hanya digunakan selama method itu

dikerjakan. Sehingga tidak perlu diakses dari luar method. Bahkan variabel yang ada di dalam blok bisa diakses dari blok itu saja.

- Variabel dalam deklarasi method – ini disebut parameter. Parameter sudah pernah dibahas panjang lebar pada bagian sub program

Deklarasi field terdiri dari 3 komponen

1. Tidak ada atau ada modifier, seperti public atau private. Sebenarnya dengan tanpa menuliskan modifier, maka kita membuat deklarasi field tersebut sebagai *default*.
2. Tipe field. Pada perkembangan pembahasan selanjutnya, tipe ini bisa saja bukan hanya tipe sederhana tetapi tipe yang kompleks.
3. Nama field. Dalam pembuatan nama, aturan penamaan harus diikuti. Dan sangat dianjurkan untuk menggunakan huruf kecil sebagai huruf pertama.



- **Akses Modifier**

Modifier pertama (paling kiri) yang digunakan menyebabkan kita bisa mengontrol apakah kelas-kelas lain mempunyai akses ke field anggota. Kita akan memfokuskan pada pembahasan public dan private saja, modifier yang lain akan dibicarakan lebih lanjut pada pembahasan mengenai pemrograman berorientasi obyek.

- Modifier public — field ini bisa diakses dari semua kelas.
- Modifier private —field ini hanya bisa diakses dalam kelas itu sendiri.

- **Tipe dan Nama Variabel**

Semua variabel harus mempunyai tipe. Kita bisa menggunakan tipe primitif seperti int, float, boolean dan lain-lain. Atau kita bisa menggunakan tipe referensi seperti larik, string atau bahkan obyek sekalipun.

Semua variabel, apakah itu field-field, variabel lokal atau parameter mengikuti aturan penamaan yang sama dengan kesepakatan yang berlaku tentang penamaan variabel

Aturan dan konvensi penamaan yang sama digunakan untuk method, nama kelas, kecuali

- Huruf pertama nama kelas harus kapital dan
- Kata pertama (atau hanya) dalam nama method harus kata kerja.

- **Menyediakan Konstruktor untuk kelas-kelas**

Sebuah kelas berisi konstruktor yang dilibatkan untuk menciptakan obyek dari desain kelas. Deklarasi konstruktor nampak seperti deklarasi method – kecuali bahwa konstruktor menggunakan nama yang sama dengan nama kelas dan tidak mempunyai tipe kembalian. Sebagai contoh, PersegiPanjang mempunyai satu konstruktor



```
 public PersegiPanjang(int
panjangAwal, int lebarAwal) {
        panjang = panjangAwal;
        lebar = lebarAwal;
    }
```

Untuk menciptakan sebuah obyek PersegiPanjang baru yang diberi nama persegi, sebuah konstruktor dipanggil dengan operator new.

```
PersegiPanjang persegi = new PersegiPanjang(30, 8);
```

new PersegiPanjang(30, 8) menciptakan ruangan dalam memori untuk obyek dan menginisialisasi fieldnya.

Meskipun PersegiPanjang hanya mempunyai satu konstruktor, kelas boleh mempunyai lebih dari satu konstruktor, termasuk konstruktor yang tanpa argumen. Kalau sebuah kelas mempunyai lebih dari satu konstruktor, berarti dia juga bersifat overloading.

- **Melewatkan informasi ke sebuah Method atau Konstruktor**

Deklarasi untuk sebuah method atau sebuah konstruktor mendeklarasikan jumlah dan tipe argumen untuk method atau konstruktor tersebut.

- **Tipe parameter**

Kita dapat menggunakan beberapa tipe data untuk sebuah parameter dari sebuah method atau konstruktor. Ini termasuk tipe data primitif, seperti double, float, dan integer,.

- **Jumlah argumen yang sembarang**

Kita dapat menggunakan sebuah konstruksi yang disebut *varargs* untuk melewatkan sejumlah sembarang nilai dke method. Kita menggunakan varargs ketika kita tidak tahu berapa banyak dari tipe tertentu argumen yang dilewatkan dalam method. Ini adalah *shortcut* untuk menciptakan sebuah larik secara manual (method sebelumnya dapat menggunakan varargs daripada sebuah larik)



Dalam sebuah method instance atau konstruktor, this adalah sebuah referensi ke obyek yang sekarang, yaitu obyek dimana method atau konstruktor dipanggil. Kita dapat mereferensi ke beberapa member dari obyek sekarang dari dalam sebuah method instance atau konstruktor dengan menggunakan kata kunci this.

## 2. CONTOH LATIHAN

Seperti pada Babsebelumnya tuliskan program berikut dengan menggunakan TextPad

```
class PersegiPanjang
{
  // kelas PersegiPanjang mempunyai
dua atribut
    public int panjang;
    public int lebar;
    public void setPanjang(int
nilaiBaru)
    {
        panjang = nilaiBaru;
    }
    public void setLebar(int
nilaiBaru)
    {
        lebar = nilaiBaru;
```

```
    }
    public int hitungLuas()
    {
        return panjang*lebar;
    }
    public int hitungKeliling()
    {
        return 2*(panjang+lebar);
    }
}
public class Panjang{
public static void main(String[] args)
{
PersegiPanjang PP = new
PersegiPanjang();
PP.setLebar(3);
PP.setPanjang(4);
System.out.println("Luas      = " +
PP.hitungLuas());
System.out.println("Keliling= " +
PP.hitungKeliling());
}
}
```

```
Hasil Output :

Luas      = 12

Keliling = 14

Press any key to continue . . .
```



## 3. LATIHAN

Buatlah program untuk menghitung keliling lingkaran, seperti berikut ini :

```
class Lingkaran{
  private double radius;
  void IsiJari(double radius)
  {
  this.radius=radius;
  }
  public double perolehPi()
  {
  return 3.14;
  }
  public double perolehKeliling()
  {
     return 2 * perolehPi() * radius;
  }
}
public class PenentuMetode{
public static void main(String[] args)
{
Lingkaran bulatan = new Lingkaran();
bulatan.IsiJari(75);
System.out.println("Keliling = " +
bulatan.perolehKeliling());
System.out.println("pi = " +
bulatan.perolehPi());
}
}
```

```
Hasil Output :
Keliling = 471.0
pi = 3.14
Press any key to continue . . .
```

Modifikasi program di atas agar mendapat keluaran Luas Lingkaran.

## 4. TUGAS

1. Buat program untuk menghitung volume balok/kubus dan menentukan apakah bangun yang dimasukkan balok atau kubus.
2. Tugas dari dosen pengampu praktikum



# BAB 16
# KELAS DAN OBYEK II

## 1. TEORI SINGKAT

Sebuah ciri khas program Java menciptakan banyak obyek, yang seperti kita ketahui, berinteraksi dengan meminta method melalui interaksi obyek-obyek tersebut, sebuah program dapat membawa bermacam-macam tugas, seperti implementasi GUI, menjalankan animasi, atau mengirimkan dan menerima informasi melalui jaringan. Sekali sebuah obyek menyelesaikan pekerjaan untuk apa obyek tersebut dibuat, sumber dayanya didaur ulang untuk untuk digunakan oleh obyek yang lain.

- **MENCIPTAKAN OBYEK**

Seperti sudah kita ketahui saat kita menciptakan sebuah obyek dari sebuah kelas, maka cetakan kelas tersebut akan ada di dalam obyek tersebut. Masing-masing pernyataan berikut diambil dari

program PersegiPanjang yang menciptakan obyek dan menuliskannya untuk variabel.

```
Titik titikAwal = new Titik(46, 188);
PersegiPanjang persegiSatu = new
PersegiPanjang(titikAwal, 200, 400);
PersegiPanjang persegiDua = new
PersegiPanjang(100, 200);
```

Baris pertama menciptakan sebuah obyek dari kelas Titik, dan baris kedua dan ketiga masing-masing menciptkan sebuah obyek dari kelas PersegiPanjang. Masing-masing dari statemen tersebut mempunyai tiga bagian:

1. **Deklarasi**: Kode yang ditulis **bold** adalah semua deklarasi variabel yang mengasosiasikan nama variabel dengan sebuah tipe dari obyek.
2. **Instansiasi**: Kata kunci new adalah operator Java yang menciptakan obyek.



3. **Inisialisasi**: Operator new diikuti oleh pemanggilan konstruktor, yang mana menginisialisasi obyek baru.

**Mendeklarasikan sebuah variabel untuk mengacu ke sebuah obyek**

Sebelumnya, kita mendeklarasikan sebuah variabel dengan menuliskan

*tipe nama*;

Ini memberitahu kompiler bahwa kita akan menggunakan nama untuk mengacu ke data dimana tipenya adalah tipe. Dengan sebuah variabel primitif, deklarasi ini juga menyediakan jumlah memori yang tepat untuk variabel.

Kita dapat juga mendeklarasikan sebuah variabel referensi pada baris sendiri. Sebagai contoh :

```
Titik titikAwal;
```

Jika kita mendeklarasikan titikAwal seperti itu, nilai tersebut akan tidak bisa ditentukan sampai sebuah obyek secara nyata diciptakan dan dituliskan. Sederhananya mendeklarasikan sebuah variabel referensi tidak menciptakan sebuah obyek. Untuk itu, kita perlu untuk menggunakan operator new seperti digambarkan pada bagian berikutnya. Kita harus menuliskan sebuah obyek ke titikAwal sebelum kita menggunakannya untuk kode kita. Jika tidak, kita akan menemukan kesalahan kompilasi.

**Instansiasi sebuah Kelas**

Operator new menginstansiasi sebuah kelas dengan alokasi memory untuk sebuah obyek baru dan mengembalikan sebuah referensi ke memori tersebut. Operator new juga melibatkan konstruktor obyek. Istilah yang mengatakan menginstansiasi sebuah kelas berarti mempunyai maksud yang sama dengan menciptakan sebuah obyek. Ketika kita menciptakan sebuah obyek, kita menciptakan



sebuah instance dari kelas, sehingga disebu dengan instansiasi sebuah kelas.

Operator new memerlukan sebuah argumen tunggal, postfix yang merupakan panggilan untuk sebuah konstruktor. Nama dari konstruktor menyediakan nama dari kelas untuk instansiasi.

Operator new mengembalikan sebuah referensi ke obyek yang diciptakan. Referensi ini biasanya dituliskan untuk sebuah variabel dari tipe yang sesuai, seperti :

```
Titik titikAwal = new titik(46, 188);
```

Referensi dikembalikan dengan operator new tidak dituliskan untuk sebuah variabel. Hal tersebut juga digunakan secara langsung dalam sebuah ekspresi. Sebagai contoh :

```
int panjang = new PersegiPanjang().panjang;
```

**Inisialisasi sebuah obyek**

Pada saat diciptakan, sebuah obyek harus diinisialisasi. Sebagai contoh akan kita lihat kode program untuk kelas Titik:

```
public class Titik {
    public int x = 0;
    public int y = 0;
   //konstruktor
    public Titik(int a, int b) {
     x = a;
     y = b;
    }
}
```

Kelas ini berisi sebuah konstruktor tunggal. Kita dapat mengenali sebuah konstruktor karena deklarasinya menggunakan nama yang sama dengan kelas dan tidak mempunyai tipe kembalian. Konstruktor dalam kelas Titik mengambil dua argumen integer, seperti dideklarasikan oleh kode



(int a, int b). Pernyataan berikut menyediakan 46 dan 88 sebagai nilai untuk argumen tersebut:

```
Titik titikAwal = new Titik(46, 88);
```

**MENGGUNAKAN OBYEK**

Sekali kita telah menciptakan obyek, kita akan bisa menggunakannya untuk sesuatu maksud tertentu. Kita mungkin perlu untuk menggunakan nilai dari salah satu fieldnya, mengubah satu diantara fieldnya atau memanggil salah satu methodnya untuk menampilkan suatu aksi.

```
int panjang = new
PersegiPanjang().panjang;
```

## 2. CONTOH LATIHAN

Seperti pada Babsebelumnya tuliskan program berikut dengan menggunakan TextPad. Program ini adalah modifikasi dari program sebelumnya.

Simpan program berikut dalam file Titik.java

```
public class Titik {
    public int x = 0;
    public int y = 0;
    //konstruktor
    public Titik(int a, int b) {
   x = a;
   y = b;
    }
}
```

Simpan program berikut dalam file PersegiPanjang1.java

```
public class PersegiPanjang1 {
    public int panjang = 0;
    public int  lebar= 0;
    public Titik awal;

    // konstruktor
    public PersegiPanjang1() {
   awal = new Titik(0, 0);
    }
    public PersegiPanjang1(Titik p) {
   awal = p;
    }
    public PersegiPanjang1(int w, int
h) {
   awal = new Titik(0, 0);
   panjang = w;
   lebar = h;
```



```
    }
    public PersegiPanjang1(Titik p,
int w, int h) {
   awal = p;
   panjang = w;
   lebar = h;
    }

    // method untuk memindahkan
persegi panjang
    public void pindah(int x, int y) {
   awal.x = x;
   awal.y = y;
    }


    // method untuk menghitung luas
persegi panjang
    public int getLuas() {
   return panjang * lebar;
    }
}
```

Simpan program berikut dalam Membuat Obyek.java

```
public class MembuatObyek {
    public static void main(String[]
args) {
     //mendeklarasikan dan menciptakan
satu obyek Titik
```

```
        Titik titikAwal = new
Titik(23, 94);
//mendeklarasikan & menciptakan 2
obyek PersegiPanjang
        PersegiPanjang1
persegiSatu=new
PersegiPanjang1(titikAwal, 100, 200);
PersegiPanjang1 persegiDua = new
PersegiPanjang1(50, 100);
 //menampilkan panjang, lebar dan luas
persegiSatu
        System.out.println("Panjang
persegiSatu: " +
                persegiSatu.panjang);
        System.out.println("Lebar
persegiSatu: " +
                persegiSatu.lebar);
        System.out.println("Luas
persegiSatu: " +
persegiSatu.getLuas());

        //mengeset posisi persegiDua
        persegiDua.awal=
persegiSatu.awal;

        //menampilkan posisi
persegiDua
        System.out.println("Posisi X
dari persegiDua: "
                + persegiDua.awal.x);
        System.out.println("Posisi Y
dari persegiDua: "
                + persegiDua.awal.y);
```



```
//memindahkan persegiDua dan
menampilkan posisi barunya
        persegiDua.pindah(40, 72);
        System.out.println("Posisi X
dari persegiDua: "
                + persegiDua.awal.x);
        System.out.println("Posisi Y
dari persegiDua: "
                + persegiDua.awal.y);
    }
}
```

## 3. LATIHAN

Buatlah program untuk membuat kelas sepeda dan macamnya.

```
public class Bicycle {

   // the Bicycle class has three
fields
    public int cadence;
    public int gear;
    public int speed;

   // the Bicycle class has one
constructor
    public Bicycle(int startCadence,
int startSpeed, int startGear) {
        gear = startGear;
        cadence = startCadence;
```

```
        speed = startSpeed;
    }

    // the Bicycle class has four
methods
    public void setCadence(int
newValue) {
        cadence = newValue;
    }

    public void setGear(int newValue)
{
        gear = newValue;
    }

    public void applyBrake(int
decrement) {
        speed -= decrement;
    }

    public void speedUp(int increment)
{
        speed += increment;
    }

}
```

Buat kelas mainnya.



## 4. TUGAS

1. Buat program tentang kelas orang dan jenisnya.
2. Tugas dari dosen pengampu praktikum